PHILIPS

PHILIPS MSX THUISCOMPUTER



HANDBOEK BASIC INSTRUCTIES

(REFERENCE MANUAL)

MSX

Inleiding

Dit "Naslagwerk voor programmeurs" ("Reference Manual") geeft een beschrijving van alle commando's, instructies en functies die MSX-BASIC kent. Om het opzoeken te vergemakkelijken zijn ze in alfabetische volgorde gerangschikt. De beschrijvingen hebben alle dezelfde indeling:

Formaat Onder "Formaat" verstaan we de manier waarop een commando, instructie of functie moet worden ingetikt, zodat MSX-BASIC begrijpt wat u bedoelt (zie onderaan op deze pagina).

Doel "Doel" beschrijft waarvoor het commando, de instructie of de functie kan worden gebruikt.

Categorie "Categorie" geeft aan of het om een commando, een instructie of een functie gaat.

Opmerkingen De "Opmerkingen" geven nadere informatie over hoe u het commando, de instructie of de functie doelmatig kunt gebruiken.
Met RETURN wordt de regeltoets bedoeld, bij sommige computers gemerkt met een omgebogen pijl. Met SHIFT wordt de hoofdlettertoets bedoeld; deze is bij sommige computers gemerkt met een pijl die naar boven wijst (niet te verwarren met een van de vier cursortoetsen).

Voorbeeld Bij alle meeste commando's, instructies en functies wordt een voorbeeld gegeven dat het gebruik illustreert.

Formaat

1 Wat achter "Formaat" met HOOFDLETTERS is aangegeven, moet letterlijk zo worden ingetoetst.
2 Wat tussen driekante haken < > is aangegeven, moet door de programmeur worden ingevuld.
3 Wat tussen rechte haken [ ] is aangegeven, is naar keuze van de programmeur en kan dus desgewenst worden weggelaten.
4 Drie punten (...) geven aan dat het voorgaande zo vaak mag worden herhaald als de lengte van de programmaregel toelaat.
5 Alle leestekens, met uitzondering van <> en [],

moeten zorgvuldig worden ingetoetst zoals ze zijn aangegeven. Dat geldt voor punten (.), komma's (,), ronde haakjes (()), dubbele punten (:), puntkomma's (;) enzovoort.

6 <X>, <Y> en <Z> geven numerieke variabelen aan.

7 <X$>, <Y$> en <Z$> geven alfanumerieke variabelen (strings) aan.

1 ABS

Formaat ABS(<X>)

Doel Geeft de absolute waarde van <X>, dus een eventueel min-teken wordt weggelaten.

Categorie Functie

Opmerkingen De uitkomst van deze functie is altijd een positief getal

---

Voorbeeld

```
10 PRINT ABS(7*(-5))
20 END
RUN
 35
```

2 ASC

Formaat ASC(<X$>)

Doel Geeft de ASCII-waarde van de eerste letter van X$.

Categorie Functie

Opmerkingen Als <X$> "leeg" is, dus geen enkel teken bevat, verschijnt de foutmelding "Illegal function call" op het scherm.

Voorbeeld

```
10 X$="TEST"
20 PRINT ASC(X$)
30 END
RUN
 84
```

3 ATN

Formaat ATN(<X>)

Doel Berekent de boogtangens van <X> in radialen.

Categorie Functie

Opmerkingen Het resultaat van deze berekening ligt tussen $-\pi/2$ en $+\pi/2$ (π (pi) = 3,1415926535)

Voorbeeld

```
10 X=5
20 PRINT ATN(X)
30 END
RUN
 1.373400766945
```

4 AUTO

Formaat AUTO [[<X>][,[<Y>]]]

Doel Het genereren van regelnummers

Categorie Commando

Opmerkingen <X> en <Y> moeten positieve, gehele getallen (integers) zijn.
Na het commando AUTO zal MSX-BASIC beginnen regelnummer <X> te genereren. Het volgende regelnummer zal <Y> hoger zijn.
Als u alleen AUTO intoetst en <X> en <Y> niet opgeeft, zal als eerste regelnummer 10 worden gekozen.
Volgt op <X> een komma, maar wordt geen waarde voor <Y> opgegeven, dan wordt de laatste waarde van <Y> aangenomen. Is <Y> nog niet eerder ingevoerd, dan is elk volgend regelnummer 10 hoger.

Wordt een regelnummer gegenereerd dat al in gebruik is, dan zal onmiddellijk achter het regelnummer een asterisk (*) verschijnen om u te waarschuwen. Voert u nu een BASIC-regel in, dan zal deze de bestaande regel met hetzelfde nummer overschrijven. Drukt u alleen RETURN in, dan blijft de oorspronkelijke programmaregel behouden.
De automatische regelnummering kan worden beëindigd door gelijktijdig CTRL en C of CTRL en STOP in te drukken. In dit geval zal de laatste regel niet aan het programma worden toegevoegd. Eindig dus altijd met een lege regel.

In het eerste voorbeeld worden de regelnummers 100, 150, 200 enz. gegenereerd.

In het tweede voorbeeld worden de regelnummers 10, 20, 30 enz. gegenereerd.

Voorbeeld 1 `AUTO 100,50`

Voorbeeld 2 `AUTO`

5 BASE

Formaat BASE(<X>)

Doel Opvragen van het eerste adres van de VDP-tabellen (Video Display Processor)

Categorie Systeem-variabele

Opmerkingen <X> moet een positief getal tussen 0 en 19 zijn. De getallen hebben de volgende betekenis:

0 = naamtabel in tekststand 1
1 = geen
2 = patroontabel in tekststand 1
3 = geen
4 = geen

5 = naamtabel in tekststand 2
6 = kleurtabel in tekststand 2
7 = patroontabel in tekststand 2
8 = sprite-attributietabel in tekststand 2
9 = sprite-patroontabel in tekststand 2

10 = naamtabel in grafische stand 1
11 = kleurtabel in grafische stand 1
12 = patroontabel in grafische stand 1
13 = sprite-attributietabel in grafische stand 1
14 = sprite-patroontabel in grafische stand 1

15 = naamtabel in grafische stand 2
16 = kleurtabel in grafische stand 2
17 = patroontabel in grafische stand 2
18 = sprite-attributietabel in grafische stand 2
19 = sprite-patroontabel in grafische stand 2

Gebruik deze variabelen alleen als u goed bekend bent met de werking van de Video Display Processor

---

Voorbeeld

```
10 SCREEN 0
20 PRINT BASE (2)
30 END
RUN
 2048
```

---

6 BEEP

Formaat BEEP

Doel Het opwekken van een kortstondige pieptoon

Categorie Instructie

Opmerkingen Deze instructie heeft hetzelfde effect als PRINT CHR$(7)
De pieptoon kan ook worden opgewekt door gelijktijdig CTRL en G in te drukken

Voorbeeld

```
10 BEEP
20 FOR I=0 TO 1000:NEXT
30 PRINT "Nu met CHR$(7)"
40 PRINT CHR$(7);
50 END
RUN
```

7 BIN$

Formaat BIN$(<X>)

Doel Geeft de binaire notatie van het decimale getal <X>

Categorie Functie

Opmerkingen Het resultaat van deze functie is een alfanumerieke waarde (string)
<X> moet een geheel getal zijn met een waarde van -32768 tot +65535

Voorbeeld

```
10 A=53
20 A$=BIN$(A)
30 PRINT A;A$
40 END
RUN
 53 110101
```

8 BLOAD

Formaat BLOAD "<randapparaat>:<bestandsnaam>" [,R][,<X>]

Doel In het computergeheugen laden van een machinetaalprogramma vanuit een randapparaat

Categorie Instructie

Opmerkingen <randapparaat> kan zijn:

CAS	=	datarecorder
A	=	diskettestation 1
B	=	diskettestation 2

<bestandsnaam> is de naam van het machinetaalprogramma

Als de toevoeging ",R" wordt gebruikt, zal het programma gaan lopen zodra het in het computergeheugen is geladen

Het programma zal in het computergeheugen worden geladen vanaf het adres dat is opgegeven toen het werd weggeschreven met het commando BSAVE, plus de waarde <X>. Als <X> niet is opgegeven, is <X> nul.

BLOAD CAS kan alleen worden gebruikt als een datarecorder op uw MSX-computer is aangesloten.

BLOAD A en BLOAD B kunnen alleen worden gebruikt als een of twee diskettestations op uw MSX-computer zijn aangesloten.

Voorbeeld

```
BLOAD "CAS:TEST",R,&H20
```

---

9 BSAVE

Formaat BSAVE "<randapparaat>:<bestandsnaam>", <X>,<Y>,[,<Z>]

Doel Het wegschrijvan van een machinetaalprogramma naar een randapparaat

Categorie Instructie

Opmerkingen <randapparaat> kan zijn:

CAS	=	datarecorder
A	=	diskettestation 1
B	=	diskettestation 2

<bestandsnaam> is de naam van het machinetaalprogramma

<X> is het geheugenadres waar het programma begint

<Y> is het geheugenadres waar het programma eindigt

<Z> is het geheugenadres waar de uitvoering van het programma moet beginnen. Als <Z> wordt weggelaten, wordt aangenomen dat de uitvoering van het programma op adres <X> moet beginnen.

BSAVE CAS kan alleen worden gebruikt als een datarecorder op uw MSX-computer is aangesloten.

BSAVE A en BSAVE B kunnen alleen worden gebruikt als een of twee diskettestations op uw MSX-computer zijn aangesloten.

Voorbeeld

```
BSAVE "CAS:TEST",&HC00,&HE0FF,&HC020
```

10 CALL

Formaat CALL <benaming>[(<A$>[,<B$>]...)]

Doel Het aanroepen van een functie in een ROM-insteekmodule

Categorie Instructie

Opmerkingen De functies van MSX-BASIC kunnen worden uitgebreid door middel van een ROM-insteekmodule. Wat de met CALL aangeroepen functie <benaming> precies inhoudt wordt bepaald door de ROM-insteekmodule.

<A$> en <B$> zijn alfanumerieke constanten, die als argumenten aan de functieaanroep kunnen worden toegevoegd.

In plaats van CALL kan ook een laag liggend streepje (_) worden gebruikt.

CALL FORMAT wordt gebruikt om een diskette te initiëren, dat wil zeggen geschikt te maken voor het opslaan van MSX-programma's, bestanden en dergelijke.
CALL FORMAT kan alleen worden gebruikt als een of twee diskettestations zijn aangesloten op de MSX-computer.

CALL SYSTEM wordt gebruikt om BASIC te verlaten en terug te keren naar MSX-DOS.
CALL SYSTEM kan alleen worden gebruikt als MSX-BASIC bij het "opstarten" van de computer is ingelezen door middel van MSX-DOS.

Voorbeeld

```
CALL TALK("A","B")

of

_TALK ("A","B")
```

11 CDBL

Formaat CDBL(<X>)

Doel Een getal omzetten in een dubbeleprecisiewaarde

Categorie Functie

Opmerkingen Het resultaat van deze functie is een getal met ten hoogste 14 cijfers, dat 8 bytes geheugenruimte in beslag neemt.

Normaal rekent MSX-BASIC automatisch met dubbele precisie. De functie CDBL heeft dan ook alleen zin als u eerder in het programma met de instructie DEFSNG de desbetreffende variabele tot enkeleprecisiewaarde hebt verklaard.

Voorbeeld

```
10 A=9/7:PRINT A
20 DEFSNG A-Z
30 A=9/7:PRINT A
40 A#=CDBL(9/7):PRINT A#
50 END
RUN
 1.2857142857143
 1.28571
 1.2857142857143
```

12 CHR$

Formaat CHR$ <X>

Doel Geeft het teken dat overeenkomt met de code <X>

Categorie Functie

Opmerkingen <X> moet een waarde tussen 0 en 255 hebben

De alternatieve tekens hebben de code 1 + de bijbehorende tekencode (zie voorbeeld)

Voorbeeld

```
10 SCREEN 1
20 FOR I=32 TO 255
30 PRINT CHR$(I);
40 NEXT:PRINT:PRINT
50 FOR I=65 TO 95
60 PRINT CHR$(1)+CHR$(I);
70 NEXT
80 END
RUN
```

13 CINT

Formaat CINT(<X>)

Doel Zet een numerieke variabele of een expressie om in een geheel getal (integer)

Categorie Functie

Opmerkingen De uitkomst van deze functie is een getal met ten hoogste 5 cijfers, zonder cijfers achter de decimale punt. Het getal neemt 2 bytes geheugenruimte in beslag.

Als X < -32768 of als X > 32767, verschijnt de foutmelding "Overflow in .." op uw scherm.

Voorbeeld

```
10 A%=CINT(9/7)
20 PRINT A%
30 END
RUN
 1
```

---

14 CIRCLE

Formaat CIRCLE [STEP](<X>,<Y>),<Z>,[,<XX>] [,<YY>][,<ZZ>] [,<XXX>]

Doel Het tekenen van een cirkel of een ellips op het scherm in grafische stand 1 of 2

Categorie Instructie

Opmerkingen <X> is de X-coördinaat van het middelpunt van de cirkel; <X> moet een integer zijn met een waarde tussen 0 en 255.

<Y> is de Y-coördinaat van het middelpunt van de cirkel; <Y> moet een integer zijn met een waarde tussen 0 en 191.

Als het woord STEP wordt gebruikt, gelden de waarden van <X> en <Y> ten opzichte van de cursorpositie. In dit geval mogen <X> en <Y> negatieve integers zijn.

<Z> is de straal van de cirkel; <Z> moet een integer zijn met een waarde tussen 0 en 32767.

<XX> is de kleurcode en geeft de kleur van de cirkel aan; <XX> moet een integer zijn met een waarde van 0 tot 15. De kleurcodes hebben de volgende betekenis:

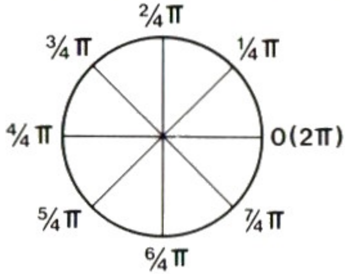
0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Als geen kleurcode wordt opgegeven, zal voor <XX> de laatst ingegeven voorgrondkleur worden gebruikt. De "default"-waarde van <XX> (dat wil zeggen de waarde bij het inschakelen van de computer) is 15.

<YY>, uitgedrukt in radialen, geeft het beginpunt van de cirkel aan. Als <YY> niet is opgegeven, geldt voor <YY> de waarde 0.

<ZZ>, eveneens uitgedrukt in radialen, geeft het eindpunt van de cirkel aan. Als de waarde van <ZZ> niet is opgegeven, heeft <ZZ> de waarde 2π.

<YY> en <ZZ> moeten een waarde hebben tussen 0 en 2π ($\pi = 3.1415926535898$). De betekenis van deze waarden is weergegeven in onderstaande afbeelding.



Als voor <YY> en <ZZ> negatieve waarden worden gebruikt, zullen begin- en eindpunt van de cirkel door middel van een rechte lijn met het middelpunt zijn verbonden.

<XXX> is een getal dat de verhouding aangeeft tussen horizontale en verticale as van een ellips. Als voor <XXX> geen waarde wordt opgegeven, is <XXX> = 1

Voorbeeld

```
10 SCREEN 2
20 A=3.1415926535898
30 CIRCLE (80,80),20,1,-(A*1/4),
-(A*6/4)
40 FOR I=0 TO 3000:NEXT
50 END
RUN
```

15 CLEAR

Formaat CLEAR [<X>][,<Y>]

Doel Maakt alle numerieke variabelen nul, maakt alle alfanumerieke variabelen (strings) leeg en reserveert geheugenruimte voor alfanumerieke variabelen.

Categorie Instructie

Opmerkingen <X> is een positief geheel getal (integer), dat het aantal bytes van alle alfanumerieke variabelen aangeeft. De voorgeprogrammeerde standaardwaarde van <X> is 200 bytes.

<Y> is een positief geheel getal, dat het hoogste adres aangeeft dat nog door MSX-BASIC mag worden gebruikt.

Alle eerder geprogrammeerde functies, zoals DEFINT, DEFSNG, DEFDBL, DEFSTR en DEFUSR, worden door CLEAR gewist. Ook alle tabellen, die met DIM zijn gedimensioneerd, worden door CLEAR gewist.

Alle geopende bestanden worden door CLEAR gesloten.

Voorbeeld

```
10 A=10:B$="TEST"
20 PRINT A,B$
30 CLEAR
40 PRINT A,B$
50 END
RUN
 10            TEST
 0
```

16 CLOAD/CLOAD?

Formaat CLOAD [" <programmanaam> "]
CLOAD? [" <programmanaam> "]

Doel Het laden van een programma van de cassette in het computergeheugen

Categorie Commando

Opmerkingen Het programma moet in binaire vorm op de cassette zijn geschreven met het commando CSAVE.

De naam van het programma is een alfanumerieke constante, die is opgegeven toen het programma op de cassette werd weggeschreven met het commando CSAVE. Als geen programmanaam is opgegeven, wordt het eerste programma ingelezen dat op de cassette wordt aangetroffen.

Als het programma op de cassette is gevonden, zet MSX-BASIC de volgende tekst op het scherm: "FOUND: programmanaam".
Stemt de naam van een gevonden programma niet overeen met die van het gezochte programma, dan ziet u "SKIP: programmanaam" op uw scherm en zal MSX-BASIC doorgaan met zoeken.

Het commando CLOAD sluit alle eerder geopende bestanden af en verwijdert een eventueel programma uit het geheugen van de computer.
Het commando CLOAD? vergelijkt het programma op cassette met het programma in het geheugen. Als de programma's identiek zijn, antwoordt MSX-BASIC met "Ok". Zijn ze niet identiek, dan verschijnt de foutmelding "Verify error" (verificatiefout) op het scherm.

CLOAD en CLOAD? kunnen alleen worden gebruikt als op de computer een datarecorder is aangesloten.

Voorbeeld

```
CLOAD "DEMO"
```

---

17 CLOSE

Formaat CLOSE [[#] <X> [,[#] <Y> ...]

Doel Het afsluiten van een bestand (file) en het vrij maken van de geheugenbuffer die daarvoor gebruikt is

Categorie Instructie

Opmerkingen De instructie OPEN wordt gebruikt om een bestand te openen en de nodige bufferruimte te reserveren.

<X> en <Y> zijn de bestandsnummers waaronder het bestand met de instructie OPEN is geopend. Als u geen nummer opgeeft, worden met de instructie CLOSE alle bestanden afgesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "CAS:TEST" FOR OUTPUT AS #1
30 A$="Proefbestand"
40 PRINT #1,A$
50 CLOSE #1
60 END
RUN
```

---

18 CLS

Formaat CLS

Doel Wissen van het scherm

Categorie Instructie

Opmerkingen Deze instructie heeft hetzelfde effect als PRINT CHR$(12).
In de directe stand kunt u het scherm ook wissen door de toetsen SHIFT en HOME, of CTRL en L tegelijk in te drukken.

Na het uitvoeren van de instructie staat de cursor linksboven op het scherm.

Voorbeeld

```
10 CLS
20 PRINT "Nu met CHR$(12)"
30 FOR I=0 TO 1000:NEXT
40 PRINT CHR$(12)
50 END
RUN
```

19 COLOR

Formaat COLOR [<X>][,<Y>][,<Z>]

Doel Het definiëren van de kleuren op het scherm

Categorie Instructie

Opmerkingen <X> bepaalt de voorgrondkleur, <Y> de achtergrondkleur en <Z> de kleur van de rand in tekststand 2 en de grafische standen 1 en 2.

<X>, <Y> en <Z> zijn de nummers van de kleuren;

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Als voor <X>, <Y> en <Z> geen waarde wordt ingevoerd, gebruikt MSX-BASIC de laatst opgegeven kleur. Bij het inschakelen van de computer gebruikt MSX-BASIC voor <X>, <Y> en <Z> de waarden 15, 4 en 7 (de zogenaamde "default"-waarden).

Voorbeeld

```
10 SCREEN 1
20 COLOR 1,1,1
30 FOR I=0 TO 15
40 PRINT "Kleur";I;",1,1"
50 COLOR I,1,1
60 FOR K=0 TO 300:NEXT K
70 NEXT I
80 COLOR 1,1,1
90 FOR I=0 TO 15
100 PRINT "Kleur 1,";I;",1"
110 COLOR 1,I,1
120 FOR K=0 TO 300:NEXT K
130 NEXT I
140 COLOR 15,1,1
150 FOR I=0 TO 15
160 PRINT "Kleur 15,1,";I
170 COLOR 15,1,I
180 FOR K=0 TO 300:NEXT K
```

```
190 NEXT I
200 END
RUN
```

20 CONT

Formaat CONT

Doel Het hervatten van de programma-uitvoering nadat deze is onderbroken

Categorie Commando

Opmerkingen De uitvoering van het programma zal worden hervat in de regel waar hij werd onderbroken.

De uitvoering van een programma kan worden onderbroken met de instructie STOP of END, maar ook door gelijktijdig de toetsen CTRL en STOP in te drukken.

Wordt het programma onderbroken direct na een INPUT-instructie, dan zal het na het commando CONT worden hervat bij deze INPUT-instructie.

Het commando CONT wordt meestal gebruikt in combinatie met de instructie STOP met het doel fouten op te sporen in een programma. Als met de instructie STOP het programma is onderbroken, kan in de directe stand de inhoud van de variabelen worden gecontroleerd en zo nodig veranderd.

Voorbeeld

```
10 PRINT "TEST"
20 STOP
30 PRINT "Verder na CONT"
40 END
RUN
TEST
Break in 20
Ok
CONT (zelf intoetsen in de directe
stand)
Verder na CONT
```

21 COPY

Formaat COPY "<randapparaat>:<naam1>"TO"
<randapparaat>:[<naam2>"]

Doel Het kopiëren van een bestand op diskette

Categorie Instructie

Opmerkingen <randapparaat> kan zijn:
A = diskettestation 1
B = diskettestation 2

<naam1> is de naam van het bestand dat moet worden gekopieerd.
<naam2> is de naam van het gekopieerde bestand.
Als alles na TO wordt weggelaten, wordt het besstand onder dezelfde naam op de andere diskette gekopieerd.

De instructie COPY kan alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
COPY "A:TEST" TO "B:TEST2"
```

---

22 COS

Formaat COS(<X>)

Doel Berekening van de cosinus van <X> in radialen

Categorie Functie

Opmerkingen Geen

Voorbeeld

```
10 X=5
20 PRINT COS(X)
30 END
RUN
 .28366218546337
```

---

23 CSAVE

Formaat CSAVE "<programmanaam>"[,<X>]

Doel Het wegschrijven van een programma vanuit het computergeheugen naar een cassette

Categorie Commando

Opmerkingen Het programma wordt in binaire vorm op de cassette geschreven. Het kan weer worden ingelezen met het commando CLOAD.

<programmanaam> is een alfanumerieke constante. Deze naam moet weer worden gebruikt bij het inlezen van het programma met het commando CLOAD.

<X> geeft de snelheid aan waarmee het programma wordt overgebracht naar de datarecorder (Engels; baud rate). <X> moet 1 of 2 zijn;

1 = 1200 baud
2 = 2400 baud

Als <X> niet is opgegeven, wordt de laatst gekozen snelheid gebruikt. Bij het inschakelen van de computer is <X> = 1 (1200 baud).

De overdrachtsnelheid kan ook worden ingesteld met de instructie SCREEN.

CSAVE kan alleen worden gebruikt als op de computer een datarecorder is aangesloten.

Voorbeeld

```
CSAVE "DEMO"
```

24 CSNG

Formaat CSNG(<X>)

Doel Zet een numerieke variabele of een expressie om in een getal met enkele precisie

Categorie Functie

Opmerkingen De uitkomst van deze functie is een getal met ten hoogste 6 cijfers. Dit getal neemt 4 bytes geheugenruimte in beslag.

Voorbeeld

```
10 A!=CSNG(9/7)
20 PRINT A!
30 END
RUN
 1.28571
```

25 CSRLIN

Formaat CSRLIN

Doel Het bepalen van de Y-coördinaat van de cursor (d.w.z. het regelnummer)

Categorie Functie

Opmerkingen De uitkomst van deze functie is een getal van 0 tot en met 23. De bovenste regel heeft het nummer nul.

Deze functie CSRLIN kan alleen worden gebruikt in tekststand 1 en 2.

Zie ook POS(0)

Voorbeeld

```
5 SCREEN 0
10 LOCATE 10,20
20 PRINT CSRLIN
30 END
RUN
            20
```

---

26 CVI/CVS/CVD

Formaat CVI(<X$>)
CVS(<Y$>)
CVD(<Z$>)

Doel Het omzetten van alfanumerieke waarden (strings) in numerieke waarden

Categorie Functie

Opmerkingen Numerieke waarden moeten eerst worden omgezet in strings voordat ze kunnen worden weggeschreven naar een "random file" op diskette (een willekeurig toegankelijk bestand). Zie hiervoor de functies MKI$, MKS$ en MKD$. Bij het teruglezen moeten deze strings weer worden omgezet in numerieke waarden met de functie CVI, CVS of CVD.

CVI zet een string van 2 bytes om in een integer.

CVS zet een string van 4 bytes om in een getal met enkele precisie.

CVD zet een string van 8 bytes om in een getal met dubbele precisie.

CVI, CVS en CVD kunnen alleen worden gebruikt als een of twee diskettestations zijn aangesloten op de computer.

Voorbeeld `Zie FIELD`

---

27 DATA

Formaat DATA <constante> [,<constante> ...]

Doel Het opslaan van constanten

Categorie Instructie

Opmerkingen DATA-instructies mogen op elke plaats in het programma worden opgenomen. Een DATA-instructie mag net zo veel constanten bevatten als er in een programmaregel kunnen. De constanten moeten worden gescheiden door komma's.
In een programma mag een onbeperkt aantal DATA-instructies worden opgenomen.

Met de instructie READ kan de informatie in de DATA-instructies worden gelezen. De volgorde bij het lezen wordt bepaald door de regelnummers. De informatie kan worden beschouwd als een doorlopende lijst van constanten, ongeacht het aantal constanten per DATA-instructie en de plaats waar deze in het programma staan.
De lijst van constanten mag zowel alfanumerieke als numerieke constanten bevatten (geen expressies). Alfanumerieke constanten moeten tussen aanhalingstekens worden geplaatst als ze beginnen of eindigen met spaties en als zij een of meer komma's of dubbele punten bevatten.

Het type variabele in de READ-instructie moet corresponderen met het type constante in de DATA-instructie, zoals in het onderstaande voorbeeld.

De constanten in een DATA-instructie kunnen vanaf het begin van het programma of vanaf een bepaalde programmaregel (opnieuw) worden gelezen na het uitvoeren van de instructie RESTORE.

Voorbeeld

```
10 DATA 123,ABC,48.5
20 READ A%,A$,A!
30 PRINT A%,A$,A!
40 RESTORE
50 READ A%:PRINT A%
60 END
```

```
RUN
 123          ABC
 48.5
 123
```

28 DEF FN

Formaat DEF FN <naam>[(<variabele>[,<variabele>...])] = <formule>

Doel Het definiëren van een functie door de gebruiker

Categorie Instructie

Opmerkingen <naam> is de naam waaronder de functie kan worden aangeroepen en moet een "legale" naam zijn (d.w.z. geen naam die voor MSX-BASIC is gereserveerd).
Als het om een alfanumerieke functie gaat, moet het laatste teken van de naam het dollarteken ($) zijn. De lengte van de instructie wordt slechts beperkt door de maximale lengte van een programmaregel.

<variabele> is de variabele in de formule die bij het aanroepen van de functie moet worden vervangen door een waarde. Worden twee of meer variabelen gebruikt, dan correspondeert de eerste variabele achter de functienaam met de eerste variabele in de formule, de tweede variabele achter de naam met de tweede variabele in de formule enzovoort.

Een variabele in de formule mag ook voorkomen in een lijst van variabelen na de <naam> van de functie. In dat geval wordt deze variabele vervangen door de gedefinieerde waarde als de functie wordt aangeroepen. Anders wordt de laatste waarde gebruikt die aan de variabele is gegeven.

Als het type van de functienaam (numeriek of alfanumeriek) niet in overeenstemming is met de aard van de formule, verschijnt de foutmelding "Type mismatch" op het scherm.

Voorbeeld

```
10 DEF FNC(A,B)=A*2+B*3
20 DEF FND$(I)=CHR$(1)+CHR$(I)
30 X=5:Z=2
40 E=FNC(X,Z)
50 PRINT E
60 FOR J=0 TO 2000:NEXT
70 SCREEN 1
80 FOR J=65 TO 95
```

```
90 PRINT FND$(J)
100 NEXT J
110 END
RUN
```

29 DEFINT/DEFSNG/DEFDBL/DEFSTR

Formaat
DEFINT <letter1>[,<letter2>...]
DEFSNG <letter1>[,<letter2>...]
DEFDBL <letter1>[,<letter2>...]
DEFSTR <letter1>[,<letter2>...]

Doel Definiëren van het type variabele

Categorie Instructie

Opmerkingen <letter1> en <letter2> mogen elke willekeurige letter van het alfabet zijn, bij voorbeeld "K", maar ook een reeks letters, bij voorbeeld "K-P" Alle variabelen die met een dergelijke letter beginnen worden beschouwd als variabele van het gedefinieerde type:
DEFINT K definieert de variabelen die met de letter K beginnen als integer;
DEFSNG K definieert de variabelen die met de letter K beginnen als variabele met enkele precisie;
DEFDBL K definieert de variabelen die met de letter K beginnen als variabele met dubbele precisie;
DEFSTR K definieert de variabelen die met de letter K beginnen als stringvariabele (alfanumerieke variabele).

Wordt in een programma aan een variabele, die eerder is gedefinieerd met DEFxxx, een ander type toegekend (met %, !, # of $), dan vervalt de eerder gegeven definitie.

Voorbeeld

```
10 DEFINT I
20 DEFSNG J
30 DEFDBL K
40 DEFSTR L-N
50 I=1.6:PRINT I
60 J=1.6:PRINT J
70 K=1/3:PRINT K
80 L="ABC":M="GHF":N="KLM"
90 PRINT L;M;N
100 END
RUN
 1
```

```
 .33333333333333
ABCGHFKLM
```

30 DEFUSR

Formaat DEFUSR[<X>]=<Y>

Doel Specificatie van het beginadres van een machinetaal-subroutine

Categorie Instructie

Opmerkingen <X> is een integer tussen 0 en 9. Als <X> niet is vermeld, wordt <X> =0 aangenomen.

<Y> is het adres waar de uitvoering van de subroutine moet beginnen; dit hoeft niet noodzakelijkerwijs het eerste geheugenadres van de routine te zijn.

De instructie DEFUSR mag in een programma een onbeperkt aantal keren worden gebruikt om een nieuw beginadres te definiëren.

De machinetaalroutine wordt aangeroepen met de functie USR.

Voorbeeld `zie USR`

31 DELETE

Formaat DELETE [<X>][-<Y>]

Doel Het verwijderen van programmaregels

Categorie Commando

Opmerkingen <X> en <Y> zijn regelnummers. Bij opgave van <X> en <Y> worden alle regels met nummers van <X> tot en met <Y> verwijderd (voorbeeld 1).

Geeft u alleen -<Y> op, dan worden alle regels vanaf de eerste programmaregel tot en met <Y> verwijderd (voorbeeld 2).

Geeft u alleen <X> op, dan wordt alleen de regel met nummer <X> verwijderd (voorbeeld 3).

Geeft u noch <X>, noch <Y> op, dan krijgt u de foutmelding "Illegal function call".

<X> en <Y> moeten bestaande regelnummers zijn, anders krijgt u de foutmelding "Illegal function call".

In het eerste voorbeeld worden de regels 50 tot en met 100 verwijderd, in het tweede voorbeeld de regels 0 tot en met 100 en in het derde voorbeeld alleen regel 100.

Voorbeeld 1

```
DELETE 50-100
```

Voorbeeld 2

```
DELETE -100
```

Voorbeeld 3

```
DELETE 100
```

---

32 DIM

Formaat DIM <variabele>(<X>[,<Y>[,<Z>]])

Doel Het dimensioneren van een tabel met variabelen

Categorie Instructie

Opmerkingen Zonder DIM is het aantal elementen in een variabelentabel maximaal 11 (0 tot en met 10).

<variabele> is de naam van de variabele.

<X> is het aantal elementen in de eerste dimensie.
<Y> is het aantal elementen in de tweede dimensie.
<Z> is het aantal elementen in de derde dimensie.

Wordt een variabele gebruikt met een index die groter is dan het aantal elementen dat is gedimensioneerd met de instructie DIM, dan verschijnt de foutmelding "Subscript out of range" op het scherm.

Met de instructie DIM wordt geheugenruimte voor de hele tabel gereserveerd.

Een gedimensioneerde variabele kan in een programma niet opnieuw worden gedimensioneerd, tenzij de tabel eerst wordt gewist met de instructie ERASE.

Voorbeeld

```
10 FOR I=0 TO 10
20 PRINT A(I)
30 NEXT
40 ERASE A:DIM A(25):DIM A$(25)
50 FOR I=0 TO 25
60 A(I)=I:A$(I)=CHR$(65+I)
70 NEXT
80 FOR I=0 TO 25
90 PRINT A(I);A$(I)
100 NEXT
110 END
RUN
```

33 DRAW

Formaat DRAW <X$>

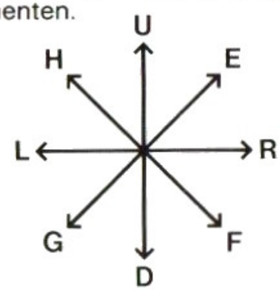
Doel Het trekken van een lijn in grafische stand 1 en 2

Categorie Instructie

Opmerkingen <X$> is een alfanumerieke constante (string), opgebouwd uit een reeks subcommando's voor richting, kleur, schaalfactor, hoek en dergelijke van de lijn die moet worden getrokken. De volgende subcommando's kunnen worden gebruikt:

1 Richting en lengte
De volgende subcommando's hebben tot gevolg dat een lijn wordt getrokken vanaf de positie van de cursor in één van de acht windrichtingen. De lijn heeft de lengte <X>, uitgedrukt in beeldelementen.

U<X> = noord (Up)
D<X> = zuid (Down)
L<X> = west (Left)
R<X> = oost (Right)
E<X> = noordoost
F<X> = zuidoost
G<X> = zuidwest
H<X> = noordwest



Deze subcommando's mogen vooraf worden gegaan door het subcommando B. De cursor zal dan naar de aangegeven positie gaan, zonder dat een lijn wordt getrokken. Het subcommando N, voorafgaand aan bovenstaande subcommando's, heeft tot gevolg dat de cursor terugkeert naar het begin van de getrokken lijn.

2 Coördinaten
Met het subcommando M<X>,<Y> kan vanaf de plaats van de cursor een lijn worden getrokken naar een plaats op het scherm, aangegeven met de absolute coördinaten <X> en <Y>.
Wordt <X> voorafgegaan door een plus- of een minteken (+ of -), dan is de X-coördinaat relatief, dat wil zeggen ten opzichte van de plaats van de cursor. Hetzelfde geldt voor de Y-coördinaat.
Na het uitvoeren van het subcommando bevindt de

cursor zich op het eindpunt van de getrokken lijn.

Het subcommando M mag vooraf worden gegaan door het subcommando B. Dan zal de cursor naar de aangegeven positie gaan, zonder dat een lijn wordt getrokken. Het subcommando N, voorafgaand aan het subcommando M, heeft tot gevolg dat de cursor terugkeert naar het begin van de getrokken lijn.

3 Kleur
Het subcommando C<X> geeft de kleur aan waarin de lijn moet worden getrokken. <X> is de kleurcode, die de kleur van de lijn aangeeft; <X> moet een integer zijn met een waarde van 0 tot 15. De kleurcodes hebben de volgende betekenis;

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Komt het subcommando C<X> niet voor in de string <X$>, dan zal de laatst ingegeven voorgrondkleur worden gebruikt. De "default"-waarde van <X> (dat wil zeggen de waarde bij het inschakelen van de computer) is 15 (wit)

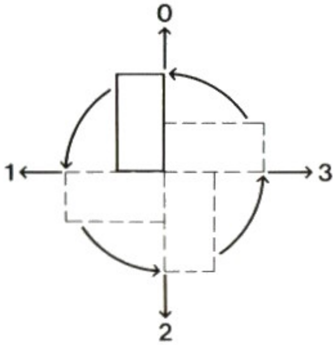
4 Schaalfactor
Het subcommando S<X> geeft de schaalfactor aan. De schaalfactor is gelijk aan <X> gedeeld door 4, waarbij <X> een integer tussen 1 en 255 moet zijn. De lijnlengten die zijn opgegeven met een van de subcommando's U, D, L, R, E, F, G en H of met het subcommando M<X>,<Y>, worden vermenigvuldigd met de schaalfactor.

Komt het subcommando S<X> niet voor in de alfanumerieke constante <X$>, dan wordt de laatst opgegeven schaalfactor gebruikt. Bij het inschakelen van de computer wordt <X> op de waarde 4 gezet, overeenkomend met schaalfactor 1.

5 Hoek
Het subcommando A<X> kan worden gebruikt om de richting van de lijn over een hoek van 90, 180 of 270 graden te verdraaien.

<X> is een integer tussen 0 en 3 en heeft de volgende betekenis:
0 = 0°
1 = 90°
2 = 180°
3 = 270°

Komt het subcommando A<X> niet voor in de alfanumerieke constante <X$>, dan wordt de laatst opgegeven hoek gebruikt. Bij het inschakelen van de computer wordt <X> op de waarde 0 gezet, overeenkomend met een verdraaiing van 0°.



6 Variabelen in plaats van constanten
De subcommando's voor kleur, richting en dergelijke kunnen worden opgenomen in de stringconstante <X$>, zoals in regel 100 van het programmavoorbeeld. Ze kunnen echter ook vooraf worden opgenomen in een stringvariabele <Z$>, zoals in regel 50 van het programmavoorbeeld, om vervolgens te worden uitgevoerd met het subcommando X<Z$>;. Zie hiervoor regel 60. Vergeet niet de puntkomma.

Iets dergelijks geldt voor de numerieke constanten die in <X$> voorkomen. Deze constanten kunnen worden gesubstitueerd door numerieke variabelen, mits die worden voorafgegaan door het isgelijkteken (=). In regel 20 wordt aan de variabelen X1 en X2 een waarde toegekend, en in regel 30 worden deze waarden toegekend aan het subcommando M. Vergeet ook hier de puntkomma niet.

Voorbeeld

```
10 SCREEN 2
20 X1=80:X2=90
30 DRAW "BM=X1;,=X2;"
40 FOR I=0 TO 3
50 AB$="A"+STR$(I)+"S"+STR$(I*5+4)+
"C1U20L10D20R10"
60 DRAW "XAB$;"
70 FOR J=0 TO 300:NEXT J
80 NEXT I
90 FOR J=0 TO 3000:NEXT J
100 DRAW "A0S8E23C10"
110 GOTO 110
120 END
RUN
```

34 DSKF

Formaat DSKF(<X>)

Doel Het opvragen van het aantal vrije clusters op de diskette in het aangegeven diskettestation

Categorie Functie

Opmerkingen <X> is het nummer van het diskettestation.

DSKF kan alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
10 PRINT DSKF(1)
20 END
RUN
```

35 END

Formaat END

Doel Het afbreken van de uitvoering van een programma

Categorie Instructie

Opmerkingen Bij het uitvoeren van de instructie END worden alle geopende bestanden afgesloten en keert MSX-BASIC terug naar de directe stand.

De instructie END mag op elke plaats in een programma worden opgenomen.

De instructie END wordt meestal als laatste instructie in een programma opgenomen, maar dat is niet verplicht.

Voorbeeld

```
10 INPUT "Voer een getal in ";K
20 IF K=0 THEN END
30 PRINT K*K
40 GOTO 10
RUN
```

36 EOF

Formaat EOF(<X>)

Doel Onderzoeken of het einde van een sequentieel bestand is bereikt

Categorie Functie

Opmerkingen <X> is het bestandsnummer waaronder het bestand is geopend met de instructie OPEN. Het bestand moet zijn geopend in de INPUT-stand (zie de instructie OPEN).

De uitkomst van deze functie is -1 als het einde van het bestand is bereikt. Anders zal de uitkomst 0 zijn.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "CAS:DEMO" FOR INPUT AS #1
30 IF EOF(1)=-1 GOTO 70
40 INPUT #1,A$
50 PRINT A$
60 GOTO 30
70 CLOSE #1
80 END
RUN
```

---

37 ERASE

Formaat ERASE <variabele1>[,<variabele2>...]

Doel Het verwijderen van tabellen uit het geheugen van de computer

Categorie Instructie

Opmerkingen <variabele1> en <variabele2> zijn de namen van de variabelen waarvoor met de instructie DIM een tabel is gedimensioneerd.

Een tabel voor variabele <X> of <X$> moet eerst met de instructie ERASE <X> of ERASE <X$> worden gewist voordat voor dezelfde variabele met DIM een nieuwe tabel kan worden gedimensioneerd. Anders verschijnt de foutmelding "Redimensioned array" op uw scherm.

Voorbeeld

```
10 DIM A(15)
20 DIM B(13)
30 ERASE A,B
40 DIM A(80)
50 DIM B(20)
60 END
RUN
```

---

38 ERL

Formaat ERL

Doel Het opvragen van het nummer van de programmaregel waarin door MSX-BASIC een fout is gevonden

Categorie Functie

Opmerkingen In een programma kan een foutafhandelingsroutine worden opgenomen. De instructie ON ERROR GOTO maakt dat MSX-BASIC bij het constateren van een fout naar de foutafhandelingsroutine springt. De functie ERL in deze routine geeft als uitkomst het nummer van de regel waarin de fout is geconstateerd.

Constateert MSX-BASIC een fout terwijl u in de directe stand werkt, dan geeft de functie ERL het regelnummer 65535.

Voorbeeld

```
10 ON ERROR GOTO 50
20 A=25:PRINT A
30 B=A/0
40 END
50 PRINT "Fout in regel";ERL
60 RESUME NEXT
RUN
 25
Fout in regel 30
```

39 ERR

Formaat ERR

Doel Het opvragen van het nummer van de fout die door MSX-BASIC is gevonden

Categorie Functie

Opmerkingen In een programma kan een foutafhandelingsroutine worden opgenomen. De instructie ON ERROR GOTO maakt dat MSX-BASIC bij het constateren van een fout naar de foutafhandelingsroutine springt. De functie ERR in deze routine geeft als uitkomst het codenummer van de fout die is geconstateerd.

Voor de codenummers van de foutmeldingen: zie appendix A van de gebruiksaanwijzing.

Voorbeeld

```
10 ON ERROR GOTO 50
20 A=25:PRINT A
30 B=A/0
40 END
50 PRINT "Fout nummer";ERR
60 RESUME NEXT
RUN
 25
Fout nummer 11
```

40 ERROR

Formaat ERROR <X>

Doel Het definiëren van uw eigen foutcodes en het simuleren van fouten

Categorie Instructie

Opmerkingen <X> is de foutcode. <X> moet een integer zijn met een waarde van 0 tot en met 255.

Is <X> een bestaande foutcode, dan zal de instructie ERROR<X> de desbetreffende fout simuleren en de daarmee corresponderende foutmelding op het scherm zetten. Toetst u bij voorbeeld "ERROR 10" in, dan krijgt u de foutmelding "Redimensioned array".

MSX-BASIC gebruikt de foutcodes 0 tot en met 59. De instructie ERROR<X> geeft voor alle waarden van <X> groter dan 59 en kleiner dan 256 de foutmelding "Unprintable error".

De foutcodes 60 tot 255 kunt u gebruiken om uw eigen foutmeldingen te definiëren. Op die manier kunt u zelf fouten in de uitvoering van een programma opvangen en afhandelen, zoals in het voorbeeld. Wordt in regel 20 een tekst ingevoerd die langer is dan 10 tekens, dan wordt foutcode 255 opgewekt. Het programma springt naar de foutafhandelingsroutine in regel 50 en hervat de uitvoering in regel 20.

Voorbeeld

```
10 ON ERROR GOTO 50
20 INPUT "Welke tekst";A$
30 IF LEN(A$)>10 THEN ERROR 250
40 END
50 IF ERR=250 THEN PRINT "Tekst te
lang":RESUME 20
60 ON ERROR GOTO 0
70 END
RUN
```

41 EXP

Formaat EXP(<X>)

Doel Berekenen van e tot de macht <X>

Categorie Functie

Opmerkingen De functie EXP(<X>) berekent het grondtal van de natuurlijke logaritmen (2.7182818284588), verheven tot de macht <X>.

<X> mag niet groter zijn dan 145.06286058562, anders krijgt u de foutmelding "Overflow".

Voorbeeld

```
10 FOR I=1 TO 4
20 PRINT USING "##.####";EXP(I);
LOG(EXP(I))
30 NEXT
40 END
RUN
 2.7183 1.0000
 7.3891 2.0000
20.0855 3.0000
54.5982 4.0000
```

42 FIELD

Formaat FIELD [#]<X>,<Y> AS <Y$>[,<Z> AS <Z$>...]

Doel Het indelen van de buffer voor een "random" bestand op diskette

Categorie Instructie

Opmerkingen Voordat in een programma de instructies GET en PUT kunnen worden gebruikt, moet de instructie FIELD worden uitgevoerd.
Voordat de instructie FIELD kan worden gebruikt, moet eerst met de instructie OPEN een bestand worden geopend.

<X> is het nummer van het bestand dat met de instructie OPEN is geopend.
<Y> en <Z> geven het aantal bytes aan dat voor de strings <Y$> en <Z$> in de buffer moet worden gereserveerd.

Het totale aantal bytes in de buffer mag niet groter zijn dan 256.

De instructie FIELD plaatst geen informatie in de buffer, maar reserveert alleen de nodige ruimte. Het invoeren van informatie in de buffer gebeurt met de instructies LSET en RSET.

Als met twee of meer "random" (= willekeurig toegankelijke) bestanden wordt gewerkt, kunnen verscheidene FIELD-instructies worden gebruikt om voor elk bestand de buffer in te delen.

De instructie FIELD kan alleen worden gebruikt als op de computer een of twee diskettestations zijn aangesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "A:TEST" AS #1
30 FIELD #1,2 AS N1$,4 AS N2$,8 AS
N3$,20 AS N4$
40 INPUT "A%";A%
```

```
50 INPUT "B!";B!
60 INPUT "C#";C#
70 INPUT "D$";D$
80 RSET N1$=MKI$(A%):RSET N2$=MKS$
(B!):RSET N3$=MKD$(C#):LSET N4$=D$
90 PUT #1,1
100 A%=0:B!=0:C#=0:D$=""
110 PRINT A%;B!;C#;D$
120 GET #1,1
130 A%=CVI(N1$):B!=CVS(N2$):C#=CVD
(N3$):D$=N4$
140 PRINT A%;B!;C#;D$
150 CLOSE #1
160 END
RUN
```

43 FILES

Formaat FILES [" <randapparaat> : <naam> "]

Doel Opvragen van de inhoudsopgave van een diskette

Categorie Commando

Opmerkingen <randapparaat> kan zijn:
A = diskettestation 1
B = diskettestation 2

<naam> kan de naam van een bestand of een programma zijn (Engels: file name).

Het commando FILES onderzoekt de inhoudsopgave van een diskette. Worden <randapparaat> en <naam> opgegeven, en is het desbetreffende programma of bestand aanwezig op de diskette, dan zal de naam daarvan op het scherm verschijnen. Anders verschijnt de mededeling "File not found" op het scherm.

Als <randapparaat> en <naam> worden weggelaten, worden de namen van alle programma's en bestanden op het scherm gezet die op de diskette in het geselecteerde diskettestation voorkomen.

De instructie FILES kan alleen worden gebruikt als op de computer een of twee diskettestations zijn aangesloten.

Voorbeeld

```
FILES "A:DEMO"
```

---

44 FIX

Formaat FIX(<X>)

Doel Berekening van het integer-deel van <X>

Categorie Functie

Opmerkingen Deze functie is gelijk aan SGN(<X>)*INT(ABS(<X>)).

Het verschil tussen FIX(<X>) en INT(<X>) is dat bij negatieve getallen INT afrondt op de eerstvolgende kleinere integer en FIX op de eerstvolgende grotere integer.

Bij positieve getallen is er geen verschil tussen FIX en INT.

Voorbeeld

```
10 A=-1.7345
20 PRINT FIX(A)
30 PRINT INT(A)
40 END
RUN
-1
-2
```

45 FOR-NEXT

Formaat FOR <XX> = <X> TO <Y> [STEP <Z>]

-

-

-

NEXT [<XX>[,<YY>]...]

Doel Het herhaald laten uitvoeren van instructies door middel van een programmalus

Categorie Instructie

Opmerkingen <XX> en <YY> zijn numerieke variabelen.

<XX> fungeert als teller, <X> is de beginstand van de teller en <Y> is de eindstand.

Alle programmaregels na de instructie FOR worden uitgevoerd tot aan de instructie NEXT. Dan wordt de teller verhoogd met de stapwaarde <Z>. MSX-BASIC zal nu eerst onderzoeken of de stand van de teller hoger is dan de eindstand <Y>. Is dat niet het geval, dan gaat MSX-BASIC terug naar de instructie volgend op de instructie FOR en wordt het hele proces herhaald. Met andere woorden: de programmalus wordt nogmaals doorlopen. Is de stand van de teller hoger dan de aangegeven eindstand <Y>, dan springt MAS-BASIC naar de instructie die volgt op de instructie NEXT.

Is geen stapgrootte <Z> opgegeven, dan wordt de teller elke keer dat de programmalus is doorlopen met 1 verhoogd.

Is voor de stapgrootte <Z> een negatief getal opgegeven, dan wordt de teller met de waarde <Z> verlaagd telkens als de programmalus is doorlopen. In dit geval moet de beginstand <X> hoger zijn dan de eindstand <Y>. Is de stapgrootte een positief getal, dan moet de beginstand <X> lager zijn dan de eindstand <Y>.

De programmalus zal ten minste eenmaal worden doorlopen als de beginstand van de teller <X>, vermenigvuldigd met het teken van de stapwaarde (+ of -) hoger is dan de eindstand <Y>, eveneens vermenigvuldigd met het teken van de stapwaarde.

Een FOR-NEXT-lus mag worden gebruikt binnen een andere FOR-NEXT-lus (Engels: nesting).In dat geval moeten voor de numerieke variabele <XX> verschillende namen worden gebruikt, bij voorbeeld I en J. De NEXT-instructie van de binnenste lus moet voor die van de buitenste lus worden geplaatst.

Hebben binnenste en buitenste lus hetzelfde eindpunt, dan mogen beide variabelen aan dezelfde NEXT-instructie worden toegevoegd (zie regel 70 van het voorbeeld).

Wordt de instructie NEXT niet gevolgd door de naam van de variabele, dan neemt MSX-BASIC aan dat deze NEXT betrekking heeft op de laatste FOR-instructie.

Komt MSX-BASIC een NEXT-instructie tegen zonder voorafgaande FOR-instructie, dan leidt dit tot de foutmelding "NEXT without FOR".

Voorbeeld

```
10 FOR I=0 TO 4
20 PRINT I;
30 NEXT:PRINT
40 FOR I=0 TO 5 STEP 5
50 FOR J=I TO I+2
60 PRINT J;
70 NEXT J,I:PRINT
80 END
RUN
```

---

46 FRE

Formaat FRE(0)
FRE("")

Doel Opvragen hoeveel bytes geheugenruimte nog niet door MSX-BASIC worden gebruikt

Categorie Functie

Opmerkingen FRE(0) geeft het aantal bytes geheugenruimte dat nog niet door MSX-BASIC wordt gebruikt voor het opslaan van het programma en de bijbehorende variabelen.

FRE("") geeft het aantal bytes geheugenruimte dat nog beschikbaar is voor alfanumerieke variabelen. Dit aantal kan worden veranderd met de instructie CLEAR.

Voorbeeld

```
10 CLEAR
20 PRINT FRE(0);FRE("")
30 END
RUN
```

47 GET

Formaat GET [#]<X>[,<Y>]

Doel Het in de buffer inlezen van een "record" van een "random" bestand vanaf een diskette

Categorie Instructie

Opmerkingen <X> is het nummer waaronder het willekeurig toegankelijke bestand is geopend met de instructie OPEN.

<Y> is het nummer van de "record" (dat wil zeggen een aantal bij elkaar behorende gegevens, bij voorbeeld een naam en een adres uit een adressenbestand). <Y> moet een integer zijn met een waarde tussen 0 en 32767. Is <Y> niet opgegeven, dan wordt de volgende "record" ingelezen, dat wil zeggen de record met het nummer dat 1 hoger is dan dat van de laatste ingelezen record.

Voordat "records" kunnen worden ingelezen, moet in het computergeheugen een buffer zijn gereserveerd met de instructie FIELD.

Gebruik de instructie PUT om een "record" vanuit de buffer weg te schrijven naar het willekeurig toegankelijke bestand op de diskette.

De instructie GET kan alleen worden gebruikt als op de computer een of twee diskettestations zijn aangesloten.

Voorbeeld `Zie de instructie FIELD`

---

48 GOSUB-RETURN

Formaat GOSUB <X>
-
-
-
RETURN [<Y>]

Doel Naar een subroutine springen en terugkeren naar het hoofdprogramma

Categorie Instructie

Opmerkingen <X> is het nummer van de eerste programmaregel van de subroutine.

Een subroutine mag in een programma zo dikwijls worden aangeroepen als nodig is. Het is ook mogelijk binnen een subroutine een andere subroutine aan te roepen (Engels: nesting).

De instructie RETURN maakt dat MSX-BASIC teruggaat naar de instructie die onmiddellijk volgt op de laatste GOSUB-instructie.
Staat achter RETURN een regelnummer (<Y>), dan keert MSX-BASIC terug naar de desbetreffende programmaregel.

Subroutines mogen overal in het programma worden geplaatst. Het verdient echter aanbeveling subroutines die vaak worden aangeroepen vóór het hoofdprogramma te plaatsen en subroutines die maar zelden worden aangeroepen na het hoofdprogramma.

Voorbeeld

```
10 GOSUB 90
20 PRINT "Regel 20"
30 GOSUB 90
40 PRINT "Regel 40"
50 I=1:GOSUB 90
60 PRINT "Regel 60"
70 PRINT "END"
80 END
90 PRINT"I=";I;
100 PRINT "Subroutine"
110 IF I=1 THEN RETURN 70
```

```
120 RETURN
RUN
```

49 GOTO

Formaat GOTO <X>

Doel Naar een gegeven programmaregel springen

Categorie Instructie

Opmerkingen <X> is een regelnummer. De uitvoering van het programma gaat door op deze programmaregel.

Voorbeeld

```
10 GOTO 40
20 PRINT "Subroutine"
30 RETURN
40 PRINT "Hoofdprogramma"
50 GOSUB 20
60 END
RUN
```

50 HEX$

Formaat HEX$(<X>)

Doel Berekenen van de hexadecimale waarde van het decimale getal <X>

Categorie Functie

Opmerkingen De uitkomst van deze functie is een alfanumerieke waarde.

<X> wordt zo nodig eerst afgerond tot een integer door weglating van eventuele decimalen. <X> moet liggen tussen -32768 en 65535.

Is <X> een negatief getal, dan wordt het twee-complement berekend. Dit betekent dat HEX$(-1) = HEX$(65536-1).

Voorbeeld

```
10 INPUT "Voer een getal in";A
20 A$=HEX$(A)
30 PRINT "Decimaal"A"= hexadecimaal "A$
40 GOTO 10
RUN
```

51 IF-THEN-ELSE

Formaat

IF <expressie> THEN <instructie>
[ELSE <instructie>]

IF <expressie> THEN [GOTO]<regelnummer>
[ELSE [GOTO]<regelnummer>]

IF <expressie> THEN <instructie>
[ELSE [GOTO]<regelnummer>]

IF <expressie> THEN [GOTO]<regelnummer>
[ELSE <instructie>]

Doel

Het nemen van een beslissing over de verdere programma-uitvoering, afhankelijk van de uitkomst van een expressie

Categorie

Instructie

Opmerkingen

Als de uitkomst van de expressie waar is (dat wil zeggen: ongelijk aan nul is), zal de instructie achter THEN worden uitgevoerd of zal het programma verder gaan met het achter THEN genoemde regelnummer.
Is de uitkomst van de expressie niet waar (dat wil zeggen gelijk aan nul), dan zal de instructie achter ELSE worden uitgevoerd of zal het programma verder gaan met het regelnummer dat achter ELSE is genoemd. Ontbreekt ELSE, dan gaat het programma verder met de volgende programmaregel.

Wordt achter THEN of ELSE een regelnummer genoemd, dan mag of THEN, of GOTO worden weggelaten. U hebt dus de keus tussen GOTO<regelnummer> en THEN<regelnummer>.

In een expressie mag de waarde van een variabele worden vergeleken met een constante of met de inhoud van een andere variabele. Daarbij zijn de volgende vergelijkingen mogelijk:

>	groter dan
<	kleiner dan
=	gelijk aan
< > of > <	ongelijk aan
> = of = >	groter dan of gelijk aan
< = of = <	kleiner dan of gelijk aan

Deze vergelijkingen mogen worden omgekeerd door gebruik te maken van het woord NOT.

In een expressie mogen ook verscheidene vergelijkingen worden uitgevoerd door gebruik te maken van de logische operatoren AND, OR, XOR, EQV en IMP.

Wordt in een IF-THEN-ELSE-instructie met een GOTO-instructie of een THEN-instructie verwezen naar een niet bestaand regelnummer, dan leidt dat tot de foutmelding "Undefined line number".

Een IF-THEN-ELSE-instructie mag worden gebruikt binnen een andere IF-THEN-ELSE-instructie ("nesting").

Voorbeeld

```
10 INPUT "Voer een getal in (5...25)";I
20 IF I<5 OR I>25 THEN 70 ELSE IF I<20
AND I>10  GOTO 50
30 PRINT "Getal ligt tussen 5 en 10 of
tussen 20  en 25"
40 GOTO 10
50 PRINT "Getal ligt tussen 10 en 20"
60 GOTO 10
70 PRINT "Getal is niet correct"
80 GOTO 10
RUN
```

52 INKEY$

Formaat INKEY$

Doel Opvragen van het teken dat correspondeert met een ingedrukte toets

Categorie Functie

Opmerkingen De letter, het cijfer of ander teken dat is ingetoetst verschijnt niet automatisch op het scherm.

Is geen toets ingedrukt, dan geeft INKEY$ als resultaat een lege string (""), overeenkomend met de tekencode nul.

Voorbeeld

```
10 REM Gebruik de cursortoetsen
20 SCREEN 2:COLOR 1,4,15:X=127:Y=95
30 I$=INKEY$:IF I$="" THEN 30
40 B=ASC(I$)
50 IF B<28 OR B>31 THEN 30
60 A=B-27:ON A GOTO 70,90,110,130
70 X=X+1:IF X>255 THEN X=255
80 GOTO 140
90 X=X-1:IF X<0 THEN X=0
100 GOTO 140
110 Y=Y-1:IF Y<0 THEN Y=0
120 GOTO 140
130 Y=Y+1:IF Y>191 THEN Y=191
140 PSET (X,Y)
150 GOTO 30
RUN
```

53 INP

Formaat INP(<X>)

Doel Het opvragen van een byte, afkomstig van machine-ingangspoort <X>

Categorie Functie

Opmerkingen <X> moet een integer tussen 0 en 255 zijn.

Gebruik de instructie OUT om een byte naar een machine-uitgangspoort te sturen.

Voorbeeld

```
10 A$=INKEY$:IF A$="" THEN 10
20 PRINT A$
30 A=ASC(A$)
40 OUT &H98,A
50 B$=HEX$(INP(&H98)):PRINT B$
60 PRINT:GOTO 10
70 END
```

54 INPUT

Formaat INPUT [<vraag>;]<variabele>,<variabele>...

Doel Het toekennen van een waarde aan een variabele via het toetsenbord

Categorie Instructie

Opmerkingen Als het programma een INPUT-instructie tegenkomt, zal het een vraagteken op het scherm zetten en wachten totdat de gebruiker gegevens invoert via het toetsenbord. De ingetoetste gegevens worden toegekend aan de desbetreffende variabelen. Achter één INPUT-instructie mogen verscheidene variabelennamen worden opgenomen, gescheiden door komma's. Het aantal gegevens dat de gebruiker invoert moet overeenkomen met het aantal variabelen dat achter de INPUT-instructie is vermeld.

De gebruiker mag de gegevens achter elkaar invoeren, gescheiden door komma's, maar ook elk ingevoerd gegeven bevestigen met RETURN. In het laatste geval zet MSX-BASIC voor de tweede en volgende variabelen twee vraagtekens op het scherm, zolang tot aan alle variabelen een waarde is toegekend.

Toetst u meer gegevens in dan er variabelen zijn, dan verschijnt de foutmelding "Extra ignored" (extra gegevens genegeerd) op het scherm.

<vraag> is een tekst, die automatisch voor het vraagteken op het scherm wordt gezet. Het is verstandig in <vraag> tot uitdrukking te brengen welk type gegevens (numeriek, alfanumeriek e.d.) de gebruiker moet intoetsen.

<variabele> mag zowel een numerieke als een alfanumerieke variabele zijn, mits de naam van de variabele daarmee in overeenstemming is (<X> of <X># voor dubbele-precisievariabelen, <X>$ voor stringvariabelen enz.).

Worden aan een numerieke variabele alfanumerieke gegevens toegekend, dan komt de foutmelding "Redo from start" (Opnieuw) op het scherm.

Worden aan een stringvariabele numerieke gegevens toegekend, dan worden deze beschouwd als alfanumerieke waarden.

Voorbeeld

```
10 INPUT"Voer artikel en aantal in"
;A$,A%
20 PRINT A%"stuks "A$
30 END
RUN
Voer artikel en aantal in? pennen,32
 32 stuks pennen
```

55 INPUT

Formaat INPUT # <X>,<variabele>[,<variabele>...]

Doel Het inlezen van een waarde uit een sequentieel bestand en het toekennen daarvan aan een variabele

Categorie Instructie

Opmerkingen Het verschil tussen INPUT en INPUT # is dat bij de eerste de gevraagde gegevens moeten worden ingetoetst en bij de laatste die gegevens uit een sequentieel bestand worden gelezen.

<X> is het nummer waaronder het bestand met de instructie OPEN is geopend.

Het bestand moet zijn geopend in de stand INPUT.

Ook bij INPUT # kan een reeks gegevens aan een reeks variabelen worden toegekend, waarbij de variabelennamen worden gescheiden door komma's.
Het type van de gegevens moet overeenstemmen met het type variabele (numeriek, alfanumeriek e.d.).

De gegevens worden weggeschreven naar het sequentiële bestand met de instructie PRINT #.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "CAS:TEST" FOR INPUT AS #1
30 INPUT #1,A$
40 PRINT A$
50 CLOSE #1
60 END
RUN
```

56 INPUT$

Formaat INPUT$(<X>)

Doel Het invoeren van een string ter lengte van <X> tekens via het toetsenbord

Categorie Functie

Opmerkingen <X> is het aantal letters, cijfers en andere tekens dat moet worden ingevoerd via het toetsenbord.

De ingetoetste tekens verschijnen niet op het scherm.

In het voorbeeld moet "MSX" worden ingetoetst, gevolgd door het indrukken van de toets SELECT.

De programma-uitvoering kan worden afgebroken door tegelijk CTRL en STOP in te drukken.

Voorbeeld

```
10 REM CODE = MSX + toets SELECT
20 B$="MSX"+CHR$(24)
30 PRINT"CODE:"
40 A$=INPUT$(4):PRINT
50 IF A$<>B$ THEN PRINT"Verkeerde
code":GOTO 30
60 BEEP
70 PRINT"Welkom in de wereld van MSX"
80 END
RUN
```

57 INPUT$

Formaat INPUT$(<X>,[#]<Y>)

Doel Het inlezen van een string ter lengte van <X> tekens uit bestand <Y>

Categorie Functie

Opmerkingen <X> is het aantal letters, cijfers en andere tekens dat wordt ingelezen uit het sequentiële bestand.

De ingelezen tekens verschijnen niet op het scherm.

<Y> is het nummer waaronder het bestand is geopend met de instructie OPEN.

58 INSTR

Formaat INSTR([<X>,]<Y$>,<Z$>)

Doel Bepalen van de plaats waar string <Z$> voor de eerste keer voorkomt in stringvariabele <Y$>

Categorie Functie

Opmerkingen <X> is een integer tussen 0 en 255. Als <X> wordt opgegeven, begint het zoeken van de string <Z$> op positie <X> van de alfanumerieke variabele <Y$>.

Als <Z$> niet wordt gevonden in <Y$>, is de uitkomst van de functie nul.

In het voorbeeld wordt de FOR-NEXT-lus driemaal doorlopen. De eerste keer begint het zoeken op positie 1 (I = 1) en wordt vanaf de tweede positie de string BCD gevonden. De tweede keer begint het zoeken op positie 6 (I = 1 + 5) en wordt de string BCD gevonden vanaf positie 7. De derde keer (I = 11) wordt de string BCD niet meer gevonden en is de uitkomst van de functie nul.

Voorbeeld

```
10 A$="ABCDEABCDE"
20 FOR I=1 to 15 STEP 5
30 PRINT INSTR(I,A$,"BCD")
40 NEXT
50 END
RUN
 2
 7
 0
```

59 INT

Formaat INT(<X>)

Doel Berekenen van een integer, gelijk aan of kleiner dan <X>

Categorie Functie

Opmerkingen Geen

Voorbeeld

```
10 A=-2.7
20 PRINT INT(A)
30 A=2.7
40 PRINT INT(A)
RUN
-3
 2
```

60 INTERVAL ON/OFF/STOP

Formaat INTERVAL ON
INTERVAL OFF
INTERVAL STOP

Doel Activeren van de controle op een verstreken tijdinterval

Categorie Instructie

Opmerkingen Het gewenste tijdinterval moet worden ingesteld met de instructie ON INTERVAL GOSUB (zie aldaar).

Na de instructie INTERVAL ON zal MSX-BASIC bij elke instructie controleren of het tijdinterval is verstreken. Is dat het geval, dan zal MSX-BASIC naar de subroutine gaan die met de instructie ON INTERVAL GOSUB is gegeven.

Na de instructie INTERVAL OFF zal MSX-BASIC niet langer controleren of het tijdinterval is verstreken.

Na de instructie INTERVAL STOP zal MSX-BASIC eveneens bij elke instructie controleren of het tijdinterval is verstreken. Nu zal echter niet onmiddellijk de subroutine worden uitgevoerd, die met ON INTERVAL GOSUB is gegeven.
Daarentegen zal MSX-BASIC onthouden of het tijdinterval is verstreken en de subroutine uitvoeren onmiddellijk nadat de instructie INTERVAL ON is gegeven.

Voorbeeld Zie ON INTERVAL GOSUB

---

61 KEY

Formaat KEY<X>,<X$>

Doel Het toekennen van een functie aan een van de tien functietoetsen

Categorie Instructie

Opmerkingen <X> is het nummer van de functietoets (een integer tussen 1 en 10).

<X$> is de tekst die aan de desbetreffende functietoets moet worden toegekend.
<X$> mag ten hoogste 15 tekens lang zijn.

In het algemeen zal aan de functietoetsen één van de gereserveerde woorden van MSX-BASIC worden toegekend, zodat het commando, de instructie of de functie met een enkele toetsindruk kan worden aangeroepen. U kunt de functietoetsen echter ook gebruiken om instructies, die in een programma geregeld terugkeren, met een enkele toetsindruk in te voeren in de programmaregels (zie voorbeeld 2).

Voorbeeld 1

```
10 A$="LOAD"
20 KEY 1,A$+CHR$(13)
30 END
RUN
```

Voorbeeld 2

```
10 A$="?"+CHR$(34)+"Foute keus"+CH
R$(34)
20 KEY 1,A$
RUN
100 (Druk functietoets F1 in) ?"Fo
ute keus"
LIST
10 A$="?"+CHR$(34)+"Foute keus"+CH
R$(34)
20 KEY 1,A$
100 PRINT"Foute keus"
```

---

62 KEY LIST

Formaat KEY LIST

Doel Het opvragen van een overzicht van de teksten die aan de tien functietoetsen zijn toegekend

Categorie Instructie

Opmerkingen De instructie geeft een overzicht van de functies die aan de tien functietoetsen zijn toegekend, in de volgorde van 1 tot en met 10.
Besturingstekens worden als spaties weergegeven.

Voorbeeld

```
KEY LIST
color
auto
goto
list
run
color 15,4,7
cload"
cont
list.
 run
```

---

63 KEY ON/OFF

Formaat KEY ON
KEY OFF

Doel In- en uitschakelen van de weergave van de functies, toegekend aan de functietoetsen, op regel 24 van het scherm

Categorie Instructie

Opmerkingen Met de instructie KEY OFF verdwijnt het overzicht van functietoetsen van het scherm. Met de instructie KEY ON worden ze weer zichtbaar gemaakt.

Voorbeeld

```
10 FOR I=1 TO 5
20 KEY OFF
30 FOR J=1 TO 300:NEXT
40 KEY ON
50 FOR J=1 TO 300:NEXT
60 NEXT I
70 END
RUN
```

---

64 KEY(X) ON/OFF/STOP

Formaat KEY(<X>) ON
KEY(<X>) OFF
KEY(<X>) STOP

Doel Activeren van de controle op het indrukken van een functietoets

Categorie Instructie

Opmerkingen <X> is het nummer van de functietoets (1...10).

Na de instructie KEY(X) ON zal MSX-BASIC bij elke instructie controleren of de desbetreffende functietoets is ingedrukt. Is dat het geval, dan zal MSX-BASIC naar de subroutine gaan die met de instructie ON KEY GOSUB is gegeven.

Na de instructie KEY(X) OFF zal MSX-BASIC niet langer controleren of een functietoets is ingedrukt.

Na de instructie KEY(X) STOP zal MSX-BASIC eveneens bij elke instructie controleren of een functietoets is ingedrukt. Nu zal echter niet onmiddellijk de subroutine worden uitgevoerd, die met ON KEY GOSUB is gegeven.
Daarentegen zal MSX-BASIC onthouden of een functietoets ingedrukt is geweest en de subroutine uitvoeren onmiddellijk nadat de instructie KEY(X) ON is gegeven.

Voorbeeld Zie ON KEY GOSUB

---

65 KILL

Formaat KILL "<randapparaat>:<naam>"

Doel Het wissen van een programma of een bestand op diskette

Categorie Instructie

Opmerkingen <randapparaat> kan zijn:
A = diskettestation 1
B = diskettestation 2

<randapparaat> is het diskettestation met de diskette waarop het te wissen programma of bestand staat.

De instructie KILL kan alleen worden gebruikt als op de computer een of twee diskettestations zijn aangesloten.

Voorbeeld

```
KILL "A:DEMO"
```

---

66 LEFT$

Formaat LEFT$(<X$>,<X>)

Doel Geeft een string, bestaande uit de <X> meest linkse tekens van <X$>

Categorie Functie

Opmerkingen <X> is een integer tussen 0 en 255.

Als <X> groter is dan de lengte dan <X$>, wordt de gehele inhoud van <X$> gevonden.

Als <X> nul is, is de uitkomst van de functie een lege string.

Voorbeeld

```
10 A$="BASIC"
20 FOR I=1 TO LEN(A$)
30 PRINT LEFT$(A$,I)
40 NEXT
50 END
RUN
B
BA
BAS
BASI
BASIC
```

67 LEN

Formaat LEN(<X$>)

Doel Bepalen van de lengte van de string <X$>

Categorie Functie

Opmerkingen Deze functie telt het aantal letters, cijfers en andere tekens van de string <X$>, inclusief spaties en codes.

Voorbeeld

```
10 A$="BASIC"+CHR$(13)
20 PRINT LEN(A$)
30 END
RUN
 6
```

68 LET

Formaat [LET] <X> = <Y>
[LET] <X$> = <Y$>

Doel Het toekennen van een waarde aan een variabele

Categorie Instructie

Opmerkingen Het woord LET mag worden weggelaten

Voorbeeld

```
10 LET A=10
20 LET B=A
30 LET C$="Hallo"
40 PRINT A;B;C$
50 A=10:B=A:C$="Hallo"
60 PRINT A;B;C$
RUN
 10  10 Hallo
 10  10 Hallo
```

69 LINE

Formaat LINE [[STEP](<X1>,<Y1>]-[STEP] (<X2>,<Y2>)[,<Z>] [B]F]]

Doel Het trekken van een lijn in grafische stand 1 of 2

Categorie Instructie

Opmerkingen <X1> is de X-coördinaat van het beginpunt van de lijn; het moet een integer zijn tussen 0 en 255.

<Y1> is de Y-coördinaat van het beginpunt van de lijn; het moet een integer zijn tussen 0 en 191.

Als het woord STEP wordt gebruikt, worden <X1> en <Y1> beschouwd als relatieve waarden ten opzichte van de plaats van de cursor. In dit geval mogen <X1> en <Y1> negatieve integers zijn.

Als <X1>, <Y1> of beide ontbreken, worden de laatste X- en Y-coördinaten gebruikt.

<X2> en <Y2> zijn de X- en Y-coördinaten van het eindpunt van de lijn. Ook in dit geval zijn de coördinaten relatief ten opzichte van de laatste cursor-positie als het woord STEP wordt gebruikt.

<Z> is de code van de kleur waarin de lijn moet worden getrokken. Dit moet een integer zijn met een waarde van 0 tot 15, die de volgende betekenis heeft:

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Als geen kleurcode wordt opgegeven, zal voor <Z> de laatst ingegeven voorgrondkleur worden gebruikt. De "default"-waarde van <Z> (dat wil zeggen de waarde bij het inschakelen van de computer) is 15.

Als B aan de instructie wordt toegevoegd, wordt een rechthoek getekend met de gegeven lijn als diagonaal.

Als F wordt toegevoegd, zal de rechthoek worden opgevuld met de gegeven kleur.

Voorbeeld

```
10 SCREEN 2
20 FOR I=0 TO 95 STEP 2
30 LINE (128-I,95-I)-(128+I,95+I),1,B
40 NEXT
50 FOR K=0 TO 2000:NEXT
RUN
```

70 LINE INPUT

Formaat LINE INPUT [<vraag>;]<X$>

Doel Het toekennen van een waarde aan een stringvariabele via het toetsenbord

Categorie Instructie

Opmerkingen Bij het uitvoeren van de instructie LINE INPUT zal het programma wachten totdat de gebruiker via het toetsenbord de gevraagde informatie heeft ingetoetst. Er verschijnt geen vraagteken op het scherm.

Als <vraag> is gegeven, wordt de tekst daarvan weergegeven op het scherm.

Voorbeeld

```
10 LINE INPUT "Voer een woord in";A$
20 PRINT A$
30 END
RUN
```

71 LINE INPUT

Formaat LINE INPUT # <X>,<X$>

Doel Het inlezen van een "record" uit een sequentieel bestand en toekennen daarvan aan een stringvariabele

Categorie Instructie

Opmerkingen Bij het uitvoeren van de instructie LINE INPUT zal het programma een "record" (een groep bij elkaar behorende gegevens) inlezen uit het sequentiële bestand en deze toekennen aan de stringvariabele.

<X> is het nummer waaronder het bestand met de instructie OPEN is geopend. Het bestand moet worden geopend in de INPUT-stand.

Wilt u slechts één gegeven uit het bestand lezen, dan dient u de instructie INPUT # te gebruiken.

De instructie LINE INPUT # wordt dikwijls gebruikt om een MSX-BASIC-programma in te lezen in een ander programma, alsof het een gegevensbestand is.

72 LIST

Formaat LIST [<X>[-[<Y>]]]

Doel Het weergeven van programmaregels of het hele programma op het scherm

Categorie Commando

Opmerkingen <X> en <Y> zijn regelnummers. Bij opgave van <X> en <Y> worden alle regels met nummers van <X> tot en met <Y> op het scherm weergegeven (voorbeeld 1).

Geeft u alleen -<Y> op, dan worden alle regels vanaf de eerste programmaregel tot en met <Y> weergegeven (voorbeeld 2).

Geeft u alleen <X> op, dan wordt alleen de regel met nummer <X> weergegeven (voorbeeld 3).

Geeft u noch <X>, noch <Y> op, dan wordt het hele programma, dat zich in het geheugen van de computer bevindt, weergegeven op het scherm (voorbeeld 4).

Geeft u <X> en "-" op, dan worden regel <X> en alle volgende regels op het scherm weergegeven (voorbeeld 5).

Voorbeeld 1 presenteert de regels 50 tot en met 100, voorbeeld 2 de regels 0 tot en met 100, voorbeeld 3 alleen regel 100, voorbeeld 4 het hele programma en voorbeeld 5 het hele programma vanaf regel 50.

Voorbeeld 1
```
LIST 50-100
```

Voorbeeld 2
```
LIST -100
```

Voorbeeld 3
```
LIST 100
```

Voorbeeld 4
```
LIST
```

Voorbeeld 5 LIST 50-

73 LLIST

Formaat LLIST [<X>[-[<Y>]]]

Doel Het afdrukken van programmaregels of het hele programma op de printer

Categorie Commando

Opmerkingen <X> en <Y> zijn regelnummers. Bij opgave van <X> en <Y> worden alle regels met nummers van <X> tot en met <Y> afgedrukt (voorbeeld 1).

Geeft u alleen -<Y> op, dan worden alle regels vanaf de eerste programmaregel tot en met <Y> afgedrukt (voorbeeld 2).

Geeft u alleen <X> op, dan wordt alleen de regel met nummer <X> afgedrukt (voorbeeld 3).

Geeft u noch <X>, noch <Y> op, dan wordt het hele programma, dat zich in het geheugen van de computer bevindt, door de printer afgedrukt (voorbeeld 4).

Geeft u <X> en "-" op, dan worden regel <X> en alle volgende regels afgedrukt (voorbeeld 5).

LLIST kan alleen worden gebruikt als op de computer een printer is aangesloten.

Voorbeeld 1 drukt de regels 50 tot en met 100 af, voorbeeld 2 de regels 0 tot en met 100, voorbeeld 3 alleen regel 100, voorbeeld 4 het hele programma en voorbeeld 5 het hele programma vanaf regel 50.

Voorbeeld 1

```
LLIST 50-100
```

Voorbeeld 2

```
LLIST -100
```

Voorbeeld 3

```
LLIST 100
```

Voorbeeld 4 `LLIST`

Voorbeeld 5 `LLIST 50-`

74 LOAD

Formaat LOAD "<randapparaat>:<programmanaam>"[,R]

Doel In het computergeheugen laden van een programma vanuit een randapparaat

Categorie Commando

Opmerkingen <randapparaat> kan zijn:

CAS	=	datarecorder
A	=	diskettestation 1
B	=	diskettestation 2

Het programma moet met het commando SAVE in ASCII-code op de cassette of de diskette zijn geschreven.

<programmanaam> is een stringconstante; dit moet dezelfde naam zijn waaronder het programma met het commando SAVE is weggeschreven op cassette of diskette.

Als de toevoeging ",R" wordt gebruikt, zal het programma in uitvoering worden genomen zodra het in het computergeheugen is geladen.

LOAD CAS kan alleen worden gebruikt als een datarecorder op uw MSX-computer is aangesloten.

LOAD A en LOAD B kunnen alleen worden gebruikt als een of twee diskettestations op uw MSX-computer zijn aangesloten.

Voorbeeld

```
LOAD "CAS:DEMO"
```

---

75 LOC

Formaat LOC(<X>)

Doel Het opzoeken van de actuele plaats in een gegeven bestand

Categorie Functie

Opmerkingen Het bestand moet worden geopend voordat de functie LOC kan worden gebruikt.

<X> is het nummer waarmee het bestand met de instructie OPEN is geopend.

Bij willekeurig toegankelijke ("random") bestanden geeft deze functie het nummer van de laatste "record" die is weggeschreven of ingelezen.

Bij sequentiële bestanden geeft deze functie het aantal sectoren (blokken van 256 bytes) dat is weggeschreven of ingelezen sinds het bestand werd geopend.

De functie LOC kan alleen worden gebruikt als op de computer een of twee diskettestations zijn aangesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "A:TEST" AS #1
30 FIELD #1,2 AS N1$,4 AS N2$,8 AS
N3$,20 AS N4$
40 GET #1,1
50 PRINT LOC(1)
60 CLOSE #1
RUN
```

76 LOCATE

Formaat LOCATE [<X>][,<Y>][,<Z>]

Doel De cursor naar een bepaalde plaats op het scherm sturen in tekststand 1 en 2

Categorie Instructie

Opmerkingen <X> is de X-coördinaat; dit moet een integer zijn van 0 tot 39.
Als <X> niet is opgegeven, geldt de X-coördinaat van de positie waar de cursor op dat moment staat.

<Y> is de Y-coördinaat; dit moet een integer zijn van 0 tot 23.
Als <Y> niet is opgegeven, geldt de Y-coördinaat van de positie waar de cursor op dat moment staat.

<Z> is de cursor-"schakelaar", die de waarde 0 of 1 moet hebben. Nul betekent dat de cursor zichtbaar is; 1 betekent dat de cursor onzichtbaar is op het scherm.

Als <Z> niet is opgegeven, geldt de laatste waarde van <Z>. De waarde van <Z> bij het inschakelen van de computer is 0.

Voorbeeld

```
10 SCREEN 0
20 WIDTH 36:CLS
30 FOR I=0 TO 10
40 X=INT(RND(1)*36)
50 Y=INT(RND(1)*23)
60 LOCATE X,Y:PRINT "*"
70 LOCATE 10,23:PRINT "X=";X;"Y=";Y;
80 FOR J=0 TO 1000:NEXT
90 NEXT I
100 CLS
110 END
RUN
```

77 LOF

Formaat LOF(<X>)

Doel Opvragen van de lengte van een bestand, uitgedrukt in bytes

Categorie Functie

Opmerkingen Het desbetreffende bestand moet zijn geopend voordat de functie LOF kan worden gebruikt.

<X> is het nummer waaronder het bestand met de instructie OPEN is geopend.

LOF kan alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "A:TEST" FOR INPUT AS #1
30 PRINT LOF(1)
40 CLOSE #1
50 END
RUN
```

78 LOG

Formaat LOG(<X>)

Doel Het bepalen van de natuurlijke logaritme van <X>

Categorie Functie

Opmerkingen <X> moet groter zijn dan 0

LOG(<X>) berekent de natuurlijke logaritme van <X>, dat wil zeggen met het grondtal e.

De briggse logaritme van <X>, dat wil zeggen met het grondtal 10, kan worden berekend met de formule LOG(<X>)/LOG(10).

Voorbeeld

```
10 FOR I=1 TO 4
20 PRINT LOG(I)
30 NEXT
40 END
RUN
 0
 .69314718055993
 1.0986122886681
 1.3862943611199
```

79 LPOS

Formaat LPOS(0)

Doel Opvragen van de positie van de printerkop in de printerbuffer

Categorie Functie

Opmerkingen Deze functie berekent waar het afdrukmechanisme van de printer staat volgens de gegevens in de printerbuffer van de computer. Deze positie hoeft niet overeen te komen met de werkelijke positie van het mechanisme.

LPOS kan alleen worden gebruikt als op de computer een printer is aangesloten.

Voorbeeld

```
10 FOR I=1 TO 1000
20 LPRINT "Hallo";
30 IF LPOS>38 THEN LPRINT CHR$(13)
40 NEXT
50 END
RUN
```

80 LPRINT

Formaat LPRINT [[USING <afdrukformaat> ;] <expressie> ...]

Doel Het afdrukken van informatie door de printer

Categorie Instructie

Opmerkingen De instructie LPRINT werkt op dezelfde manier als de instructie PRINT. Het enige verschil is dat de informatie niet op het scherm wordt weergegeven maar door de printer wordt afgedrukt.

LPRINT kan alleen worden gebruikt als op de computer een printer is aangesloten.

Voorbeeld

```
Zie PRINT
```

---

81 LSET

Formaat LSET <X$> = <Y$>

Doel De variabele <X$> vullen met de inhoud van de string-variabele <Y$> of de expressie <Y$>, te beginnen aan de linkerkant van <Y$>

Categorie Instructie

Opmerkingen De instructie LSET of RSET moet worden gebruikt om de gegevens, die naar een willekeurig toegankelijk bestand op diskette moeten worden weggeschreven, in de buffer te plaatsen.

Alvorens numerieke gegevens in de buffer te plaatsen moeten deze in alfanumerieke vorm worden omgezet met een van de functies MKI$, MKS$ of MKD$.

Na het lezen van deze gegevens uit de buffer moeten ze weer van alfanumerieke in numerieke vorm worden terugvertaald met een van de functies CVI, CVS of CVD.

LSET kan alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "A:TEST" AS #1
30 FIELD #1,2 AS N1$,4 AS N2$,8 AS
N3$,20 AS N4$
40 INPUT "A%";A%
50 INPUT "B!";B!
60 INPUT "C#";C#
70 INPUT "D$";D$
80 RSET N1$=MKI$(A%):RSET N2$=MKS$
(B!);RSET N3$=MKD$(C#):LSET N4$=D$
90 PUT #1,1
100 A%=0:B!=0:C#=0:D$=""
110 PRINT A%;B!;C#;D$
120 GET #1,1
130 A%=CVI(N1$):B!=CVS(N2$):C#=CVD
(N3$):D$=N4$
140 PRINT A%;B!;C#;D$
150 CLOSE #1
160 END
```

82 MAXFILES

Formaat MAXFILES = <X>

Doel Specifieren van het aantal bestanden dat gelijktijdig in een programma geopend zal zijn

Categorie Instructie

Opmerkingen <X> is het aantal bestanden; <X> moet een integer zijn met een waarde tussen 0 en 15.

Zie ook de instructie OPEN.

Voorbeeld

```
10 MAXFILES=2
20 OPEN "CAS:DEMO" FOR INPUT AS #1
30 OPEN "LPT:" FOR OUTPUT AS #2
40 INPUT #1,A$
50 PRINT #2,A$
60 CLOSE
70 END
RUN
```

83 MERGE

Formaat MERGE "<randapparaat>:<programmanaam>"

Doel Het samenvoegen van een programma op cassette of diskette en een programma in het geheugen van de computer

Categorie Commando

Opmerkingen <randapparaat> kan zijn:

CAS	= datarecorder
A	= diskettestation 1
B	= diskettestation 2

Het in te lezen programma dat moet worden samengevoegd met het programma in het geheugen, moet op cassette of diskette zijn weggeschreven in ASCII-formaat (met het commando SAVE).

Als in het ingelezen programma regels voorkomen met dezelfde nummers als het programma in het geheugen, zullen de oorspronkelijke regels worden overschreven door de regels met hetzelfde nummer uit het ingelezen programma.

MERGE CAS kan alleen worden gebruikt als een datarecorder op uw MSX-computer is aangesloten.

MERGE A en MERGE B kunnen alleen worden gebruikt als een of twee diskettestations op uw MSX-computer zijn aangesloten.

Voorbeeld

```
MERGE "CAS:DEMO"
```

84 MID$ (1)

Formaat MID$(<X$>,<X>[,<Y>])

Doel Geeft een alfanumerieke expressie, bestaande uit <Y> tekens uit de string <X$>, te beginnen op de <X>e positie van <X$>

Categorie Functie

Opmerkingen <X> en <Y> moeten integers zijn met een waarde tussen 0 en 255.

Als <Y> niet is opgegeven, of als het aantal tekens rechts van <X> kleiner is dan aangegeven is in <Y>, worden alle tekens rechts van <X> in de expressie opgenomen.

Als X>LEN(X$), dus als <X> voorbij het laatste teken van <X$> ligt, geeft MID$ een lege expressie.

Zie ook **instructie** MID$.

Voorbeeld

```
10 A$="BASIC MSX COMPUTER"
20 PRINT MID$(A$,7,3)
30 END
RUN
MSX
```

85 MID$ (2)

Formaat MID$(<X$>,<X>[,<Y>])=<Y$>

Doel Het vervangen van een deel van een alfanumerieke variabele door een andere alfanumerieke variabele of een alfanumerieke constante

Categorie Instructie

Opmerkingen De tekens in <X$>, te beginnen op positie <X>, worden vervangen door de string <Y$>.

Als <Y> is opgegeven, worden <Y> tekens van <X$> vervangen door even zovele tekens uit <Y$>.

Is <Y> niet opgegeven, dan worden alle tekens van <Y$> gebruikt, voor zover de lengte van <X$> dat toelaat.

Zie ook de **functie** MID$.

Er is dus een functie MID$, waarmee een aantal tekens uit een string kan worden gelezen, en een instructie MID$, waarmee een aantal tekens in een string kan worden vervangen door andere tekens.

Voorbeeld

```
10 A$="ABCDEFG"
20 PRINT A$
30 MID$(A$,4)="XYZ"
40 PRINT A$
50 END
RUN
ABCDEFG
ABCXYZG
```

86 MKI$/MKS$/MKD$

Formaat MKI$(<X>)
MKS$(<Y>)
MKD$(<Z>)

Doel Het omzetten van numerieke waarden in alfanumerieke waarden (strings)

Categorie Functie

Opmerkingen Numerieke waarden, die moeten worden weggeschreven naar een "random file" (een willekeurig toegankelijk bestand), moeten worden omgezet in alfanumerieke vorm voordat ze in de buffer worden geplaatst.

MKI$ zet een integer om in een string van 2 bytes.

MKS$ zet een getal met enkele precisie om in een string van 4 bytes.

MKD$ zet een getal met dubbele precisie om in een string van 8 bytes.

MKI$, MKS$ en MKD$ kunnen alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "A:TEST" AS #1
30 FIELD #1,2 AS N1$,4 AS N2$,8 AS
N3$,20 AS N4$
40 INPUT "A%";A%
50 INPUT "B!";B!
60 INPUT "C#";C#
70 INPUT "D$";D$
80 RSET N1$=MKI$(A%):RSET N2$=MKS$
(B!);RSET N3$=MKD$(C#):LSET N4$=D$
90 PUT #1,1
100 A%=0:B!=0:C#=0:D$=""
110 PRINT A%;B!;C#;D$
120 GET #1,1
130 A%=CVI(N1$):B!=CVS(N2$):C#=CVD
(N3$):D$=N4$
```

```
140 PRINT A%;B!;C#;D$
150 CLOSE #1
160 END
RUN
```

87 MOTOR ON/OFF

Formaat MOTOR [ON]
MOTOR [OFF]

Doel Het veranderen van de status van de schakelaar die de motor van de datarecorder in- en uitschakelt

Categorie Instructie

Opmerkingen De instructie MOTOR, zonder toevoeging van ON of OFF, verandert de status van de motorschakelaar. Is de motor van de datarecorder ingeschakeld, dan zal hij dus worden uitgeschakeld en omgekeerd.

Wordt ON toegevoegd, dan wordt de motor ingeschakeld. De toevoeging OFF maakt dat de motor wordt uitgeschakeld.

MOTOR kan alleen worden gebruikt als op de computer een datarecorder is aangesloten.

Voorbeeld

```
10 FOR I=1 TO 10
20 MOTOR
30 FOR J=1 TO 500:NEXT
40 NEXT
50 END
RUN
```

88 NAME

Formaat NAME "<randapparaat>:<bestandsnaam 1>" AS "<bestandsnaam 2>"

Doel Het veranderen van de naam van een bestand op diskette

Categorie Instructie

Opmerkingen <randapparaat> kan zijn:
A = diskettestation 1
B = diskettestation 2

<bestandsnaam 1> is de naam waaronder het bestand op diskette is geschreven, <bestandsnaam 2> is de nieuwe naam voor dat bestand.

NAME kan alleen worden gebruikt als een of twee diskettesations op de computer zijn aangesloten.

Voorbeeld

```
NAME "A:TEST" AS "HELP"
```

89 NEW

Formaat NEW

Doel Het verwijderen van een programma uit het geheugen

Categorie Commando

Opmerkingen Het commando NEW wordt gewoonlijk gebruikt om het geheugen van de computer te wissen voordat met een nieuw programma wordt begonnen.

Voorbeeld

```
NEW
```

---

90 OCT$

Formaat OCT$(<X>)

Doel Berekenen van de octale waarde van het decimale getal <X>

Categorie Functie

Opmerkingen De uitkomst van deze functie is een alfanumerieke waarde.

<X> wordt zo nodig eerst afgerond tot een integer door weglating van eventuele decimalen. <X> moet liggen tussen -32768 en 65535.

Is <X> een negatief getal, dan wordt het twee-complement berekend. Dit betekent dat OCT$(-1) = OCT$(65536-1).

Voorbeeld

```
10 INPUT "Voer een getal in";A
20 A$=OCT$(A)
30 PRINT "Decimaal"A"= octaal "A$
40 GOTO 10
RUN
```

91 ON ERROR GOTO - RESUME

Formaat ON ERROR GOTO <X>
-
-
-
RESUME [<Y>]

Doel Het afhandelen van fouten door middel van een routine en terugkeren naar het hoofdprogramma

Categorie Instructie

Opmerkingen <X> is het nummer van de eerste regel van de foutafhandelingsroutine.

Als de instructie ON ERROR GOTO is gegeven, zal MSX-BASIC bij elke fout naar de eerste regel van de foutafhandelingsroutine springen.
ERR bevat dan het nummer van de opgetreden fout en ERL bevat het regelnummer waarin de fout is opgetreden.

Is de instructie ON ERROR GOTO 0 gegeven, dan springt MSX-BASIC niet naar een foutafhandelings -routine, maar verschijnt de gebruikelijke fout melding op het scherm.

Ontdekt MSX-BASIC een fout tijdens het uitvoeren van een foutafhandelingsroutine, dan verschijnt de gebruikelijke foutmelding op het scherm.

Met de instructie RESUME kan na het uitvoeren van de foutafhandelingsroutine worden teruggekeerd naar het hoofdprogramma.

<Y> is het nummer van de regel waar de uitvoering van het hoofdprogramma moet worden hervat na het uitvoeren van de foutafhandelingsroutine. Is <Y> niet gegeven of is <Y> =0, dan wordt de uitvoering van het programma hervat in de regel waar de fout is opgetreden.

Wordt het woord NEXT gebruikt in plaats van <Y>, dan wordt de uitvoering van het hoofdprogramma hervat in de regel volgend op die waarin de fout optrad.

Een gedefinieerde foutafhandelingsroutine wordt ook aangeroepen in de directe stand, met uitzondering van "syntax errors".

92 ON GOSUB

Formaat ON <expressie> GOSUB <X>[,<Y>...]

Doel De uitvoering van het programma afleiden naar een subroutine, afhankelijk van de expressie

Categorie Instructie

Opmerkingen <X> en <Y> zijn de eerste programmaregels van de desbetreffende subroutines.

De waarde van de expressie bepaalt naar welke van de gegeven subroutines MSX-BASIC zal springen. Is die waarde bij voorbeeld 3, dan zal een sprong worden gemaakt naar de derde subroutine.

Is de waarde van de expressie geen geheel getal, dan zal die waarde worden afgerond.

De waarde van de expressie mag niet groter zijn dan 255 en mag niet negatief zijn.

Voorbeeld

```
10 FOR I=1 TO 3
20 ON I GOSUB 50,70,90
30 NEXT I
40 END
50 PRINT "I ="I"Subroutine 50"
60 RETURN
70 PRINT "I ="I"Subroutine 70"
80 RETURN
90 PRINT "I ="I"Subroutine 90"
100 RETURN
RUN
I = 1 Subroutine 50
I = 2 Subroutine 70
I = 3 Subroutine 90
```

93 ON GOTO

Formaat ON <expressie> GOTO <X>[,<Y>...]

Doel De uitvoering van het programma afleiden naar een gegeven regenummer, afhankelijk van de expressie

Categorie Instructie

Opmerkingen <X> en <Y> zijn de nummers van programmaregels.

De waarde van de expressie bepaalt naar welke van de gegeven programmaregels MSX-BASIC zal springen. Is die waarde bij voorbeeld 3, dan zal een sprong worden gemaakt naar de regel die als derde is genoemd.

De waarde van de expressie mag niet groter zijn dan 255 en mag niet negatief zijn.

Voorbeeld

```
10 FOR I=1 TO 3
20 ON I GOTO 50,70,90
30 NEXT I
40 END
50 PRINT "I ="I"Regelnummer 50"
60 GOTO
70 PRINT "I ="I"Regelnummer 70"
80 GOTO
90 PRINT "I ="I"Regelnummer 90"
100 GOTO
RUN
I = 1 REGELNUMMER 50
I = 1 REGELNUMMER 70
I = 1 REGELNUMMER 90
```

---

94 ON INTERVAL GOSUB

Formaat ON INTERVAL = <X> GOSUB <Y>

Doel Het instellen van een tijdinterval en het aangeven van een subroutine waar naartoe moet worden gesprongen als dat interval is verstreken

Categorie Instructie

Opmerkingen <X> is het aantal seconden, vermenigvuldigd met 50.

<Y> is het nummer van de eerste regel van de tijdinterval-subroutine.

Voordat de intervalroutine kan worden uitgevoerd, moet het afvangen van de instructie ON INTERVAL GOSUB worden geactiveerd met de instructie INTERVAL ON. Daarna zal MSX-BASIC automatisch met tussenpozen van X/50 seconden de gegeven subroutine uitvoeren.

Bij het uitvoeren van een foutafhandelingsroutine zal MSX-BASIC automatisch de instructie INTERVAL OFF uitvoeren.

Voorbeeld

```
10 ON INTERVAL=300 GOSUB 60
20 INTERVAL ON
30 FOR I=0 TO 10000:NEXT
40 INTERVAL OFF
50 END
60 K=K+6:PRINT K"seconden"
70 RETURN
RUN
 6 seconden
 12 seconden
```

95 ON KEY GOSUB

Formaat ON KEY GOSUB <X>[,<Y>...]

Doel Aangeven welke subroutine moet worden uitgevoerd als een van de functietoetsen wordt ingedrukt

Categorie Instructie

Opmerkingen <X> en <Y> zijn de nummers van de eerste regels van de desbetreffende subroutines.
<X> is het eerste regelnummer van de subroutine die moet worden uitgevoerd als functietoets 1 wordt ingedrukt, <Y> het nummer als functietoets 2 wordt ingedrukt enz.

Voordat de functietoets-routine kan worden uitgevoerd, moet het afvangen van de instructie ON KEY GOSUB worden geactiveerd met de instructie KEY(X) ON. Daarna zal MSX-BASIC automatisch de gegeven subroutine uitvoeren als de desbetreffende functietoets wordt ingedrukt.

Bij het uitvoeren van een foutafhandelingsroutine zal MSX-BASIC automatisch de instructie KEY(X) OFF uitvoeren.

Voorbeeld

```
10 CLS
20 LOCATE 5,5:PRINT "F1=Subroutine 1"
30 LOCATE 5,7:PRINT "F3=Subroutine 2"
40 LOCATE 5,9:PRINT "F5=Einde"
50 ON KEY GOSUB 80,,100,,120
60 KEY(1) ON:KEY(3) ON:KEY(5) ON
70 GOTO 70
80 LOCATE 10,11:PRINT "Subroutine 1"
90 RETURN
100 LOCATE 10,11:PRINT "Subroutine 2"
110 RETURN
120 LOCATE 10,11:PRINT "Einde"
130 END
RUN
```

96 ON SPRITE GOSUB

Formaat ON SPRITE GOSUB <X>

Doel Aangeven welke subroutine moet worden uitgevoerd als twee sprites met elkaar botsen

Categorie Instructie

Opmerkingen <X> is het nummer van de eerste regel van de botsings-subroutine.

Voordat de routine kan worden uitgevoerd, moet het afvangen van de instructie ON SPRITE GOSUB worden geactiveerd met de instructie SPRITE ON. Daarna zal MSX-BASIC automatisch de gegeven subroutine uitvoeren als twee sprites met elkaar botsen.

Bij het uitvoeren van een foutafhandelingsroutine zal MSX-BASIC automatisch de instructie SPRITE OFF uitvoeren.

Voorbeeld

```
10 DATA 60,66,165,129,165,153,66,60
20 DATA 60,120,219,255,255,219,102,60
30 A$=""
40 FOR I=1 TO 8
50 READ A:A$=A$+CHR$(A)
60 NEXT
70 B$=""
80 FOR I=1 TO 8
90 READ A:B$=B$+CHR$(A)
100 NEXT
110 SCREEN 2,1:COLOR 15,4,1
120 ON SPRITE GOSUB 210
130 SPRITE$(0)=A$:SPRITE$(1)=B$
140 SPRITE ON
150 A=INT(RND(1)*256)
160 B=INT(RND(1)*256)
170 FOR I=0 TO 191
180 PUT SPRITE 0,(A,I),1
190 PUT SPRITE 1,(B,191-I),15
200 NEXT:GOTO 140
210 SPRITE OFF
220 PLAY "L4CDEFEDCREFGAGFER"
230 PUT SPRITE 0,(0,208)
```

```
240 PUT SPRITE 1,(0,208)
250 I=191:RETURN
RUN
```

97 ON STOP GOSUB

Formaat ON STOP GOSUB <X>

Doel Aangeven welke subroutine moet worden uitgevoerd als de toetsen CTRL en STOP gelijktijdig worden ingedrukt

Categorie Instructie

Opmerkingen <X> is het nummer van de eerste regel van de subroutine.

Voordat de subroutine kan worden uitgevoerd, moet het afvangen van de instructie ON STOP GOSUB worden geactiveerd met de instructie STOP ON. Daarna zal MSX-BASIC automatisch de gegeven subroutine uitvoeren als CTRL en STOP gelijktijdig worden ingedrukt.

Bij het uitvoeren van een foutafhandelingsroutine zal MSX-BASIC automatisch de instructie STOP OFF uitvoeren.

Voorbeeld

```
10 ON STOP GOSUB 50
20 STOP ON
30 INPUT A$
40 IF A$="Einde" THEN STOP OFF:END ELSE
GOTO 30
50 PRINT "Tik 'Einde' in; druk daarna
op RETURN":RETURN
RUN
```

98 ON STRIG GOSUB

Formaat ON STRIG GOSUB <X>[,<Y>...]

Doel Aangeven welke subroutine moet worden uitgevoerd als een van de actietoetsen van de spelregelaars wordt ingedrukt

Categorie Instructie

Opmerkingen <X> en <Y> zijn de nummers van de eerste regels van de desbetreffende subroutines.
<X> is het eerste regelnummer van de subroutine die moet worden uitgevoerd als de spatiebalk wordt ingedrukt, <Y> het nummer als de actietoets van spelregelaar 1 wordt ingedrukt enz.

Voordat de functietoets-routine kan worden uitgevoerd, moet het afvangen van de instructie ON STRIG GOSUB worden geactiveerd met de instructie STRIG(X) ON. Daarna zal MSX-BASIC automatisch de gegeven subroutine uitvoeren als de desbetreffende actietoets wordt ingedrukt.

Bij het uitvoeren van een foutafhandelingsroutine zal MSX-BASIC automatisch de instructie STRIG(X) OFF uitvoeren.

Voorbeeld

```
10 CLS
20 ON STRIG GOSUB 50
30 STRIG(0) ON
40 GOTO 40
50 LOCATE 5,5
60 PRINT "Spatiebalk ingedrukt"
70 FOR I=1 TO 300:NEXT
80 LOCATE 5,5:PRINT SPC(25)
90 RETURN
RUN
```

99 OPEN

Formaat OPEN "<randapparaat>:<bestandsnaam>" [FOR <modus>] AS [#]<X> [LEN=<Y>]

Doel Het reserveren van een buffer voor invoer en uitvoer en het aangeven van de werkwijze (modus) voor de buffer

Categorie Instructie

Opmerkingen <randapparaat> kan zijn:

CRT	= beeldscherm in tekststand
GRP	= beeldscherm in grafische stand
LPT	= printer
CAS	= cassetterecorder
A	= diskettestation 1
B	= diskettestation 2

<bestandsnaam> is de naam waaronder het desbetreffende bestand is geregistreerd op het randapparaat. <bestandsnaam> hoeft niet te worden gebruikt als het randapparaat het beeldscherm of de printer is omdat naar deze randapparaten alleen uitvoer mogelijk is.

<modus> of werkwijze geeft aan of de gegevens vanaf het randapparaat in de buffer moeten worden geplaatst (INPUT) of vanuit de buffer naar het randapparaat moeten worden gestuurd (OUTPUT).
FOR <modus> moet alleen worden opgegeven bij INPUT van en OUTPUT naar een sequentieel bestand.

Er zijn drie modi mogelijk:

INPUT	= lezen uit een sequentieel bestand
OUTPUT	= schrijven naar een sequentieel bestand
APPEND	= schrijven na het einde van een bestaand sequentieel bestand

Wordt FOR <modus> weggelaten, dan neemt MSX-BASIC aan dat het om een "random file" (willekeurig toegankelijk bestand) gaat. In dit geval zal een nieuw bestand worden aangemaakt als op de diskette in het gekozen diskettestation geen bestand met de naam <bestandsnaam> aanwezig is.

Niet elke combinatie van randapparaat en modus is

mogelijk. Hieronder een overzicht van de combinaties die wel mogelijk zijn:

Randapparaat	OUTPUT	INPUT	APPEND	niet vermeld
CRT	●			
GRP	●			
LPT	●			
CAS	●	●		
A	●	●	●	●
B	●	●	●	●

<X> is het bestandsnummer zolang het bestand geopend is. Dit nummer wordt gebruikt om bij andere in- en uitvoerinstructies het bestand aan te duiden, zonder elke keer de volledige naam van dat bestand te moeten definiëren; voorbeelden: INPUT #1, PRINT #1 en CLOSE #1.

<Y> is een integerexpressie die, als hij wordt gebruikt, de "record"-lengte voor willekeurig toegankelijke bestanden ("random files") bepaalt. De "default"-waarde van <Y> bij het inschakelen van de computer is 256.

Een bestand moet altijd worden geopend voordat gegevens kunnen worden weggeschreven of ingelezen.

Het aantal bestanden dat gelijktijdig geopend kan zijn wordt bepaald met de instructie MAXFILES.

OPEN LPT kan alleen worden gebruikt als op de computer een printer is aangesloten.

OPEN CAS kan alleen worden gebruikt als op de computer een datarecorder is aangesloten.

OPEN A en OPEN B kunnen alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
10 SCREEN 2:COLOR 15,4,7
20 OPEN "GRP:" FOR OUTPUT AS #1
30 LINE (32,32)-(120,120),6,B
```

```
40 CIRCLE (120,120),56,1
50 PRESET (40,8)
60 PRINT #1,"Tekst op grafisch scherm"
70 FOR I=1 TO 2000:NEXT
80 END
RUN
```

100 OUT

Formaat OUT <X>,<Y>

Doel Het zenden van de waarde <Y> naar machine-uitgangspoort <X>

Categorie Instructie

Opmerkingen <X> en <Y> moeten integers tussen 0 en 255 zijn.

Gebruik de functie INP om een byte van een machine-ingangspoort op te vragen.

Voorbeeld

```
10 A$=INKEY$:IF A$="" THEN 10
20 PRINT A$
30 A=ASC(A$)
40 OUT &H98,A
50 B$=HEX$(INP(&H98)):PRINT B$
60 PRINT:GOTO 10
70 END
```

101 PAD

Formaat PAD(<X>)

Doel Opvragen van de status van een toetsenbordje

Categorie Functie

Opmerkingen Via de aansluitingen voor spelregelaars kunnen op de computer een of twee toetsenbordjes worden aangesloten.

<X> is een integer met een waarde tussen 0 en 7.

Heeft <X> een waarde tussen 0 en 3, dan wordt aangenomen dat het toetsenbordje is aangesloten op de aansluiting voor spelregelaar nummer 1.
Heeft <X> een waarde tussen 4 en 7, dan wordt aangenomen dat het toetsenbordje is aangesloten op de aansluiting voor spelregelaar nummer 2.

Is <X> gelijk aan 0 of 4, dan wordt de status van het toetsenbordje opgevraagd. Het resultaat van deze functie is -1 als een toets is ingedrukt en 0 als nog geen toets is ingedrukt.

Is <X> gelijk aan 1 of 5, dan wordt de X-coördinaat gegeven.
Is <X> gelijk aan 2 of 6, dan wordt de Y-coördinaat gegeven.

Is <X> gelijk aan 3 of 7, dan wordt de status van de schakelaar op het toetsenbordje gegeven. Het resultaat van deze functie is -1 als de schakelaar is ingedrukt en 0 als de schakelaar niet is ingedrukt.

PAD kan alleen worden gebruikt als op de computer een toetsenbordje is aangesloten.

Voorbeeld

```
10 SCREEN 2
20 AA=0
30 IF PAD(0)=0 THEN 20
40 X=PAD(1):Y=PAD(2)
50 IF AA=0 THEN PSET(X,Y) ELSE LINE
-(X,Y)
```

```
60 AA=1
70 GOTO 30
RUN
```

102 PAINT

Formaat PAINT [STEP](<X>,<Y>),<Z>[,<XX>]

Doel Het opvullen van een grafische figuur met kleur in de grafische standen 1 en 2

Categorie Instructie

Opmerkingen <X> is de X-coördinaat van het beginpunt op het scherm; <X> moet een integer zijn met een waarde tussen 0 en 255.

<Y> is de Y-coördinaat van het beginpunt op het scherm; <X> moet een integer zijn met een waarde tussen 0 en 191.

Als het woord STEP wordt gebruikt, worden <X> en <Y> beschouwd als coördinaten met de positie van de cursor als oorsprong. In dit geval mogen <X> en <Y> ook negatieve getallen zijn.

<Z> is de code van de kleur waarmee de figuur moet worden opgevuld. <XX> is de kleurcode van de contourlijn, dat wil zeggen de lijn die de figuur insluit.

In grafische stand 1 moet de contourlijn dezelfde kleur hebben als het oppervlak van de figuur. In dit geval hoeft <XX> niet te worden vermeld.
In grafische stand 2 mogen de contourlijn en het oppervlak van de figuur een verschillende kleur hebben.

<Z> en <XX> moeten integers zijn met een waarde van 0 tot 15. De kleurcodes hebben de volgende betekenis:

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Voorbeeld

```
10 SCREEN 2:COLOR 15,4,7
20 CIRCLE (80,80),20,8
30 PAINT (80,80),8
40 FOR I=1 TO 2000:NEXT
50 SCREEN 3:COLOR 15,4,7
60 LINE (10,10)-(100,100),8,B
70 PAINT (45,45),2,8
80 FOR I=1 TO 2000:NEXT
90 END
RUN
```

103 PDL

Formaat PDL (<X>)

Doel Opvragen van de status van een spelregelaar

Categorie Functie

Opmerkingen Op uw MSX-computer kunnen een of twee spelregelaars ("game paddles" of "joy sticks") worden aangesloten. "Game paddles" zijn uitgerust met een regelbare weerstand en zijn meestal voorzien van een draaiknop die elke willekeurige stand tussen de twee eindstanden kan innemen. "Joy sticks" zijn daarentegen uitgerust met 6 schakelaars, waarvan er telkens één kan worden gesloten door een stuurknuppel te bewegen of een toets in te drukken.

Bij gebruik van "game paddles" levert de functie PDL(<X>) een getal op met een waarde van 0 tot 255. <X> is een integer met een waarde van 1 tot 12.
Als <X> een oneven waarde heeft (1, 3, 5, 7, 9 of 11) neemt MSX-BASIC aan dat de spelregelaar met aansluiting 1 is verbonden.
Als <X> een even waarde heeft (2, 4, 6, 8, 10 of 12) neemt MSX-BASIC aan dat de spelregelaar met aansluiting 2 is verbonden.

Bij gebruik van "joy sticks" is de waarde van PDL(<X>) 255 bij geopende en 0 bij gesloten schakelaar. De oneven waarden van <X> corresponderen met de schakelaars van de "joy stick" die met aansluiting 1 is verbonden. De even waarden van <X> corresponderen met de schakelaars van de "joy stick" die met aansluiting 2 is verbonden.

PDL kan alleen worden gebruikt als op de computer een of twee spelregelaars zijn aangesloten.

Voorbeeld 1

```
10 REM Programma voor "game paddles"
20 PRINT PDL(1),PDL(2)
30 GOTO 20
```

Voorbeeld 2

```
10 REM Programma voor "joy sticks"
20 FOR I=1 TO 12
30 A=PDL(I):IF A=255 THEN NEXT ELSE
PRINT I:NEXT
40 GOTO 20
```

104 PEEK

Formaat PEEK(<X>)

Doel Het opvragen van de inhoud van een geheugenpositie

Categorie Functie

Opmerkingen Deze functie geeft de inhoud van een geheugenadres in de vorm van een decimaal getal met een waarde van 0 tot 255.

<X> is het geheugenadres. <X> moet een waarde hebben van -32768 tot 65535.

Als <X> een negatieve waarde heeft, dan wordt het twee-complement gebruikt. Dit betekent dat PEEK(-1) = PEEK(65536-1).

Het onderstaande voorbeeld laat zien hoe dat programma in het geheugen van uw computer staat.

De instructie POKE wordt gebruikt om een geheugenplaats met een gespecificeerde waarde te vullen.

Voorbeeld

```
10 SCREEN 0:WIDTH 36:CLS:KEY OFF
20 G=PEEK(&H8001):IF G=0 OR G=255 THEN
I=&H8001  ELSE I=&HC001
30 FOR J=I TO I+&H100:A=PEEK(J)
40 A$="00"+HEX$(A)
50 PRINT RIGHT$(A$,2)" ";
60 NEXT
70 END
RUN
```

---

105 PLAY (1)

Formaat PLAY <X$>[,<Y$>][,<Z$>]

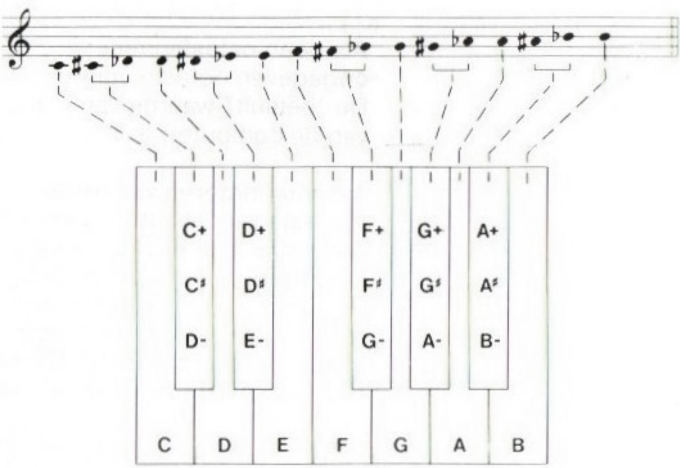
Doel Het weergeven van muziek

Categorie Instructie

Opmerkingen <X$>, <Y$> en <Z$> zijn alfanumerieke constanten, opgebouwd uit subcommando's. <X$> is de eerste, <Y$> de tweede en <Z$> de derde stem.

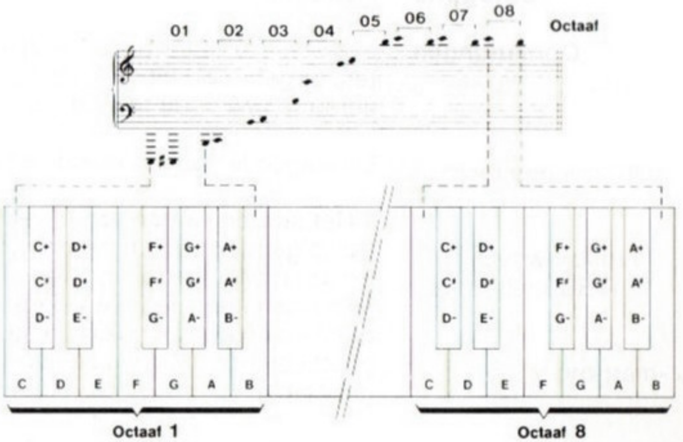
De volgende subcommando's kunnen worden gebruikt:

1 Het spelen van de tonen van een toonladder
De volgende subcommando's zullen één van de tonen van de toonladder produceren: A, B, C, D, E, F en G. Een plusteken " + " of een kruis " # " achter de letter geeft een halve toon verhoging, een minteken "-" (mol) geeft een halve toon verlaging, zoals hieronder is uitgebeeld:



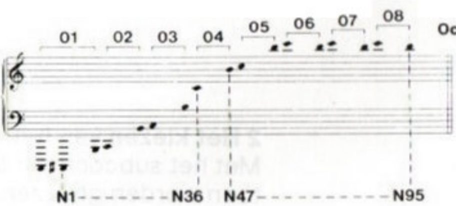
2 Het kiezen van het octaaf
Met het subcommando "O<X>" kan het octaaf van de toon worden gekozen. O is de hoofdletter o (dus geen nul) en <X> is een integer met een waarde van 1 tot en

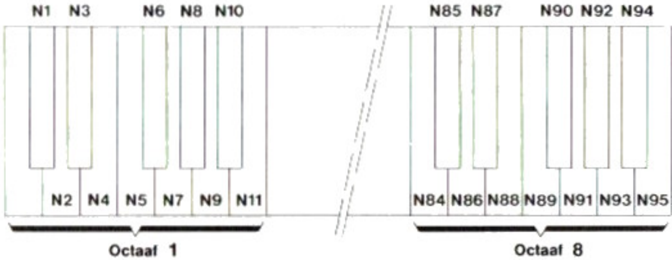
met 8.
De volgende illustratie brengt dit in beeld.



Als geen octaafnummer is gegeven, wordt het laatst opgegeven octaafnummer gebruikt.
De "default"-waarde van het octaaf (bij het inschakelen van de computer) is 4.

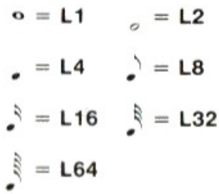
3 Het nummeren van noten
In plaats van de letters van de toonladder en de octaafnummers te gebruiken, kunt u ook de muzieknoten nummeren. Het subcommando N<X> wordt gebruikt om een noot een nummer te geven.
<X> is een integer met een waarde van 0 tot 95. Als <X> gelijk is aan 0, zal geen toon worden geproduceerd. In plaats daarvan zal een korte pauze optreden.





4 Kiezen van de lengte van een noot
De lengte van een noot wordt ingesteld met het subcommando L<X>.
<X> is een integer met een waarde van 1 tot en met 64. <X> heeft de volgende betekenis:

1 = een hele noot
2 = een halve noot
3 = eenderde noot
4 = een kwart noot
enz.



Als de lengte van de noot niet is opgegeven, wordt de laatst gespecificeerde lengte gebruikt. De "default"-waarde van een noot (bij het inschakelen van de computer) is 4 (¼ noot = 1 tel).
De lengte van de noot mag ook onmiddellijk achter de letter van de toon worden gezet, zonder de letter L. L4A is dus hetzelfde als A4.

5 Verandering van de lengte van een toon
Door onmiddellijk achter een noot een punt (.) toe te voegen, neemt de lengte van de toon toe met de helft (1,5 of 3/2 maal zo lang). Het is ook mogelijk twee of meer punten toe te voegen. Bij drie punten duurt de toon 27/8 maal zo lang.

6 Kiezen van de rust
Het subcommando R<X> wordt gebruikt om de duur van de rust te bepalen.
<X> is een integer met een waarde van 1 tot en met 64 en heeft de volgende betekenis:

1 = een volle rust
2 = een halve rust
3 = eenderde rust
4 = een kwart rust
enz.

𝄻 = R1 𝄼 = R2
𝄽 = R4 𝄾 = R8
𝄿 = R16 𝅀 = R32
𝅁 = R64

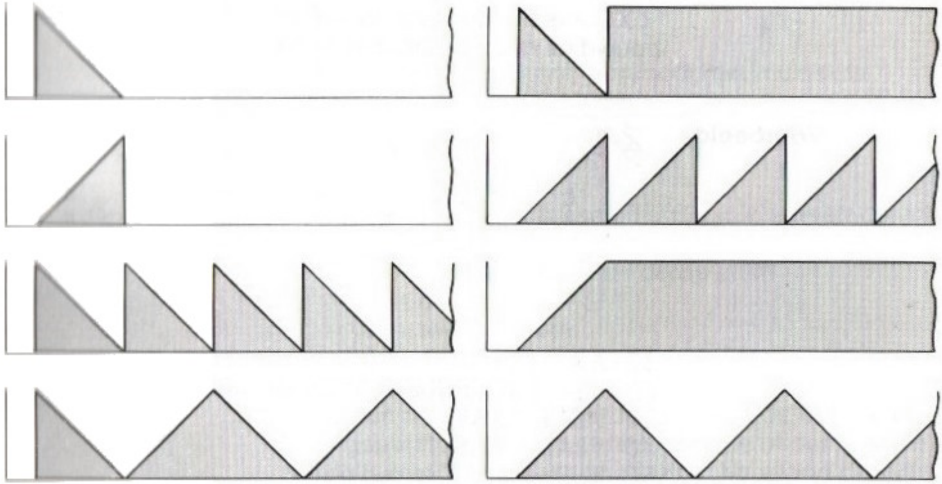
7 Kiezen van het tempo
Het tempo wordt ingesteld met het subcommando T<X>.
<X> is een integer met een waarde van 32 tot 255 en geeft het aantal kwartnoten per minuut aan. Als het tempo niet is opgegeven, wordt het laatst opgegeven tempo gebruikt.

De "default"-waarde van het tempo (bij het inschakelen van de computer) is 120.

8 Kiezen van het volume
Het volume (de geluidssterkte) kan worden gekozen met het subcommando V<X>.
<X> is een integer met een waarde van 0 tot 15.
Als het volume niet is opgegeven, wordt het laatst opgegeven volume gebruikt.

De "default"-waarde van het volume (bij het inschakelen van de computer) is 8.

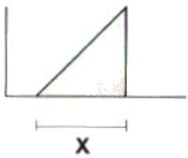
9 Kiezen van de vorm van de geluids-omhullende
De vorm van de "omhullende" van het geluid wordt ingesteld met het subcommando S<X>.
<X> is een integer met een waarde van 0 tot 15.
Deze getallen hebben de volgende betekenis:



Als de vorm van de omhullende niet is opgegeven, wordt de laatst gedefinieerde vorm gebruikt.

De "default"-waarde van de vorm (bij het inschakelen van de computer) is 1.

10 Modulatie van de vorm van de omhullende
De periodetijd van de omhullende wordt ingesteld met het subcommando M<X>.
<X> is een integer met een waarde van 1 tot 65535.
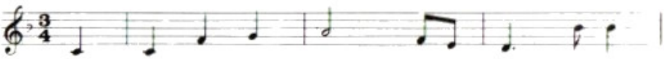


Als geen modulatie-subcommando is opgegeven, geldt het laatst opgegeven subcommando.
De "default"-waarde van de modulatie (bij het inschakelen van de computer) is 255.

11 Uitvoering van een muziekstring
De subcommando's kunnen, voordat de instructie PLAY wordt gegeven, in een string worden opgenomen. Zo'n muziekstring kan worden uitgevoerd met het subcommando X<X$>;. Vergeet niet de puntkomma.

<X$> is de naam van de stringvariabele die is opgebouwd uit muziek-subcommando's.

Voorbeeld




```
10 A$="T128"
20 B$="L4CCFGL2AL8FEL4D.L8B-L4B-"
30 PLAY "XA$;"
40 PLAY "XB$;"
50 END
RUN
```

12 Numerieke variabelen in subcommando's
Numerieke constanten in subcommando's mogen worden vervangen door numerieke variabelen. Deze variabelen moeten worden voorafgegaan door een isgelijkteken en worden gevolgd door een puntkomma.

Voorbeeld

```
10 FOR I=14 TO 84:J=I+8:K=I+2
20 PLAY "N=I;","N=K;"
30 PLAY "N=I;","N=J;","N=K;"
40 PLAY "N=J;","N=K;"
50 NEXT
RUN
```

106 PLAY (2)

Formaat PLAY(<X>)

Doel Onderzoeken of de toongenerator met muziek is geprogrammeerd.

Categorie Functie

Opmerkingen <X> is een integer tussen 0 en 3. <X> heeft de volgende betekenis:

0 = eerste, tweede en derde stem
1 = eerste stem
2 = tweede stem
3 = derde stem

Deze functie geeft de waarde -1 zolang de betrokken toongenerator (voor de eerste, tweede of derde stem) bezig is met het spelen van muziek. Na afloop daarvan krijgt de functie de waarde nul.

Wordt de functie PLAY onmiddellijk na de instructie PLAY gebruikt, dan is de uitkomst altijd -1.

Voorbeeld

```
10 A$="":BEEP
20 FOR I=1 TO 6
30 READ AA$:A$=A$+AA$
40 NEXT
50 PLAY "XA$;"
60 SCREEN 0:WIDTH 35:CLS
70 IF PLAY(0)=-1 THEN PRINT "Nu is er
muziek":  GOTO 70
80 PRINT"De muziek is opgehouden"
90 END
100 DATA CCGGAAGR
110 DATA FFEEDDCR
120 DATA GGFFEEDR
130 DATA GGFFEEDR
140 DATA CCGGAAGR
150 DATA FFEEDDCR
RUN
```

107 POINT

Formaat POINT(<X>,<Y>)

Doel Het opvragen van de kleurcode van een gegeven beeldpunt in grafische stand 1 en 2

Categorie Functie

Opmerkingen <X> is de X-coördinaat en moet een getal zijn van 0 tot en met 255.

<Y> is de Y-coördinaat en moet een getal zijn van 0 tot en met 191.

Voorbeeld

```
10 SCREEN 2:COLOR 15,1,7
20 OPEN "GRP:" FOR OUTPUT AS #1
30 FOR I=1 TO 10
40 PRESET (I*16,8)
50 C=INT(RND(1)*13)+2
60 COLOR C
70 PRINT #1,CHR$(1)+CHR$(66)
80 NEXT
90 COLOR 15
100 PRESET(40,24)
110 PRINT #1,"Kleurcodes"
120 FOR I=1 TO 10
130 C=POINT(I*16+4,12)
140 PRESET(I*16,I*8+56),4
150 PRINT #1,USING"##";C
160 NEXT
170 FOR I=0 TO 2000:NEXT
180 COLOR 15,4,7
190 END
RUN
```

---

108 POKE

Formaat POKE<X>,<Y>

Doel Het opvullen van een geheugenpositie met de waarde <Y>

Categorie Instructie

Opmerkingen Deze instructie geeft de mogelijkheid een geheugenpositie (RAM) te vullen met een decimaal getal met een waarde van 0 tot 255 of een hexadecimaal getal met een waarde van &H00 tot &HFF.

<X> is het geheugenadres. <X> moet een waarde hebben van -32768 tot 65535.

Als <X> een negatieve waarde heeft, dan wordt het twee-complement gebruikt. Dit betekent dat POKE-1,0 = POKE (65536-1),0.

De instructie PEEK wordt gebruikt om de inhoud van een geheugenplaats te bepalen.

Voorbeeld

```
10 A=PEEK(65535):PRINT A
20 POKE 65535,A
30 END
RUN
```

109 POS

Formaat POS(0)

Doel Geeft de X-coördinaat (het kolomnummer) van de cursor

Categorie Functie

Opmerkingen De uitkomst van deze functie is een getal met een waarde van 0 tot en met 39. Als de cursor helemaal links staat, is het kolomnummer 0.
De nul tussen haakjes achter POS heeft geen betekenis.

De functie POS kan alleen worden gebruikt in tekststand 1 en 2.

Zie ook CSRLIN.

PRESET

Formaat PRESET [STEP](<X>,<Y>)[,<Z>]

Doel Kleur toekennen aan een gespecificeerd beeldpunt in grafische stand 1 en 2

Categorie Instructie

Opmerkingen <X> is de X-coördinaat van het beeldpunt; <X> moet een integer zijn met een waarde tussen 0 en 255.

<Y> is de Y-coördinaat van het beeldpunt; <Y> moet een integer zijn met een waarde tussen 0 en 191.

Als het woord STEP wordt gebruikt, gelden de waarden van <X> en <Y> ten opzichte van de cursorpositie. In dit geval mogen <X> en <Y> negatieve integers zijn.

<Z> is de kleurcode; <Z> moet een integer zijn met een waarde van 0 tot 15. De kleurcodes hebben de volgende betekenis:

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Als geen kleurcode wordt opgegeven, zal voor <Z> de laatst ingegeven achtergrondkleur worden gebruikt. De "default"-waarde van <Z> (dat wil zeggen de waarde bij het inschakelen van de computer) is 15.

Zie ook PSET

Voorbeeld

```
10 SCREEN 2:COLOR 15,4,7
20 LINE(40,40)-(215,151),15,BF
30 FOR I=0 TO 1000
40 A=INT(RND(1)*173)+41
50 B=INT(RND(1)*109)+41
60 PRESET (A,B)
70 FOR K=0 TO 300:NEXT
```

```
80 NEXT I
90 END
RUN
```

111 PRINT

Formaat PRINT [[USING <printformaat> ;] <expressie> ...]

Doel Het schrijven van informatie op het scherm

Categorie Instructie

Opmerkingen Het woord PRINT zonder enige toevoeging maakt dat MSX-BASIC een lege regel afdrukt.
Wordt een expressie toegevoegd, dan wordt die op het scherm afgedrukt. Dat mag zowel een numerieke (bij voorbeeld PRINT 3*4) als een alfanumerieke expressie zijn (bij voorbeeld PRINT "Rob"). Alfanumerieke constanten moeten tussen aanhalingstekens staan.

1 Posities op het scherm
De plaats waar een expressie op het scherm verschijnt hangt af van de interpunctie tussen de verschillende expressies, dat wil zeggen de gebruikte leestekens. MSX-BASIC verdeelt het scherm in zones van 14 posities. Staat na de expressie een komma, dan verschijnt de volgende expressie vanaf het begin van de eerstvolgende zone. De komma maakt dus dat MSX-BASIC naar de volgende tabulatorstop gaat.

Staat na een expressie een puntkomma of een spatie, dan wordt de tweede expressie onmiddellijk achter de eerste op het scherm gezet.

Eindigt een PRINT-instructie met een komma of een puntkomma, dan gaat MSX-BASIC voor de volgende PRINT-instructie niet naar de volgende regel. Ontbreekt de komma of de puntkomma, dan wordt de volgende PRINT-instructie op de volgende regel uitgevoerd.

Is het aantal tekens van een expressie groter dan het aantal posities van een regel op het scherm (in te stellen met de instructie WIDTH), dan gaat MSX-BASIC automatisch verder op de volgende regel.

2 Getallen
Getallen, zoals de uitkomsten van berekeningen (dus geen getallen in de vorm van strings), worden op het scherm gevolgd door een spatie (bij voorbeeld: PRINT 2*3" =6" levert op: 6 =6). Positieve getallen worden

voorafgegaan door een spatie, negatieve getallen door een minteken.

3 PRINT-formaat
Met behulp van het woord USING kan de manier worden gekozen waarop een numerieke of alfanumerieke expressie op het scherm moet worden afgedrukt.

4 Alfanumerieke expressies
Van alfanumerieke expressies kan alleen het eerste teken worden afgedrukt met behulp van USING "!";.

Voorbeeld

```
10 A$="Europa"
20 PRINT USING"!";A$
30 END
RUN
E
```

Met USING"\\"; kunnen twee of meer tekens van een alfanumerieke string op het scherm worden gezet. Het aantal tekens is gelijk aan 2 plus het aantal spaties tussen de twee "back slashes" " \ \".

Voorbeeld

```
10 A$="Europa"
20 PRINT USING"\  \";A$
30 END
RUN
Euro
```

Met USING"& &" kunt u een string tussenvoegen op de plaats van de komma tussen twee andere strings.

Voorbeeld

```
10 A$="Ik":B$="Europa"
20 PRINT USING"& woon in &";A$,B$
30 END
RUN
```

```
Ik woon in Europa
```

---

5 Numerieke expressies

Met PRINT "#"; kan het aantal cijfers van een getal worden bepaald dat moet worden weergegeven. Is het aantal cijfers van het getal kleiner dan het aantal cijfers dat moet worden weergegeven, dan vult MSX-BASIC de voorafgaande ruimte op met spaties.

Is het aantal cijfers van het getal daarentegen groter dan het aantal cijfers dat moet worden weergegeven, dan plaatst MSX-BASIC het procentteken % voor het weergegeven getal.

Zo nodig wordt het getal door MSX-BASIC afgerond.

Voorbeeld

```
10 A=109:B=7:C=1198
20 PRINT USING"###";A,B,C
30 END
RUN
109  7%1198
```

Met "#.#" kan worden aangegeven waar de decimale punt moet worden afgedrukt. Er is maar één decimale punt toegestaan.

Op deze manier kunnen bij voorbeeld bedragen recht onder elkaar worden gezet, zodat de decimale punten in dezelfde kolom komen.

Voorbeeld

```
10 A=10.21:B=5.5:C=.245:D=3
20 PRINT TAB(10)USING"##.##";A
30 PRINT TAB(10)USING"##.##";B
40 PRINT TAB(10)USING"##.##";C
50 PRINT TAB(10)USING"##.##";D
60 END
RUN
          10.21
           5.50
           0.25
           3.00
```

USING "#,.#" kan worden gebruikt om grote getallen door middel van komma's in groepjes van drie cijfers te verdelen. Dit is dus de Amerikaanse manier.

Voorbeeld

```
10 A=107^/7:PRINT A
20 PRINT USING"########.####";A
30 PRINT USING"########,.####";A
40 END
RUN
 1428571.4285714
 1428571.4286
1,428,571.4286
```

Bij gebruik van USING "+#" worden positieve expressies voorafgegaan door een plusteken en negatieve expressies door een minteken. USING"#+" maakt dat de expressies worden gevolgd door respectievelijk een plus- of een minteken.

Voorbeeld

```
10 A=1.25:B=-1.25
20 PRINT USING"+#.##";A,B
30 PRINT USING"#.##+";A,B
40 END
RUN
+1.25-1.25
1.25+1.25-
```

USING "#-" maakt dat positieve expressies worden gevolgd door een spatie en negatieve expressies door een minteken. Het minteken mag alleen aan het einde staan, dus na het laatste hekje (#).

Voorbeeld

```
10 A=1.25:B=-1.25
20 PRINT USING"#.##-";A,B
30 END
```

```
RUN
1.25 1.25-
```

Met USING "** #" kan een expressie worden aangevuld met sterretjes als het aantal cijfers voor de decimale punt kleiner is dan met USING" # #" is gespecificeerd. De twee asterisken mogen alleen aan het begin staan, dus voor het eerste hekje #.

Voorbeeld

```
10 A=110.255:B=1.25:C=-1.25
20 PRINT USING"**###.##";A
30 PRINT USING"**###.##";B
40 PRINT USING"**###.##";C
50 END
RUN
**110.26
****1.25
***-1.25
```

USING "$#" en USING "ƒ#" heeft tot gevolg dat voor het bedrag een dollar- of een guldenteken wordt afgedrukt (het guldenteken krijgt u op het scherm met CODE + 1).

Voorbeeld

```
10 A=110.55:B=1.25:C=-1.25
20 PRINT USING"f#####.##";A
30 PRINT USING"f#####.##";B
40 PRINT USING"f#####.##";C
50 END
RUN
f  110.55
f     1.25
f    -1.25
```

USING "$$#" (dus met twee dollartekens) maakt dat onmiddellijk voorafgaand aan het bedrag een dollarteken

wordt afgedrukt.

Voorbeeld

```
10 A=110.55:B=1.25:C=-1.25
20 PRINT USING"$$#####.##";A
30 PRINT USING"$$#####.##";B
40 PRINT USING"$$#####.##";C
50 END
RUN
   $110.55
     $1.25
    -$1.25
```

Met USING"^#^^^^" kunnen getallen in exponentiële vorm worden afgedrukt.
De decimale punt mag op elke willekeurige plaats staan.
Op de eerste positie zal een spatie worden afgedrukt, tenzij is aangegeven dat een plus- of een minteken moet worden afgedrukt.

Voorbeeld

```
10 A=234.56:B=12.34:C=-12.34
20 PRINT USING"##.##^^^^";A
30 PRINT USING"#.##^^^^-";C
20 PRINT USING"+#.##^^^^";B,C
30 END
RUN
 2.35E+02
1.23E+01-
+1.23E+01-1.23E+01
```

Uit het laatste voorbeeld blijkt dat het PRINT-formaat een combinatie van formaten mag zijn. Andere voorbeelden zijn USING"$#####.##-" en USING "**$#.##-".

5 ? in plaats van PRINT
In plaats van voluit het woord PRINT kunt u ook het vraagteken ? gebruiken. MSX-BASIC zet dit automatisch om in PRINT.

Het woord LPRINT moet u echter altijd voluit intoetsen. Weliswaar zet MSX-BASIC L? op het scherm om in LPRINT, maar in de desbetreffende regel geeft dit een "syntax error".

112 PRINT

Formaat PRINT # <X>,[USING <printformaat> ;] <variabele>

Doel Het wegschrijven van informatie naar een sequentieel bestand.

Categorie Instructie

Opmerkingen <X> is het nummer waaronder het desbetreffende bestand is geopend met de instructie OPEN.
Het bestand moet zijn geopend in de stand OUTPUT.

Numerieke expressies worden door MSX-BASIC automatisch gescheiden door een komma als ze met een enkele PRINT #-instructie naar een bestand worden weggeschreven. Alfanumerieke expressies moeten door de programmeur worden gescheiden door middel van een komma.

De instructie INPUT # wordt gebruikt om gegevens uit een sequentieel bestand te lezen.

Voorbeeld 1

```
10 OPEN "CAS:DEMO" FOR OUTPUT AS 1#
20 A=110.55:B=1.25:C=-1.25
30 PRINT USING"$$#####.##";A
40 PRINT USING"$$#####.##";B
50 PRINT USING"$$#####.##";C
60 CLOSE
70 END
RUN
```

Voorbeeld 2

```
10 OPEN "CRT:" FOR OUTPUT AS 1#
20 A=110.55:B=1.25:C=-1.25
30 PRINT USING"$$#####.##";A
40 PRINT USING"$$#####.##";B
50 PRINT USING"$$#####.##";C
60 CLOSE
70 END
RUN
   $110.55
     $1.25
    -$1.25
```

---

113 PSET

Formaat PSET [STEP](<X>,<Y>)[,<Z>]

Doel Kleur toekennen aan een gespecificeerd beeldpunt in grafische stand 1 en 2

Categorie Instructie

Opmerkingen <X> is de X-coördinaat van het beeldpunt; <X> moet een integer zijn met een waarde tussen 0 en 255.

<Y> is de Y-coördinaat van het beeldpunt; <Y> moet een integer zijn met een waarde tussen 0 en 191.

Als het woord STEP wordt gebruikt, gelden de waarden van <X> en <Y> ten opzichte van de cursorpositie. In dit geval mogen <X> en <Y> negatieve integers zijn.

<Z> is de kleurcode; <Z> moet een integer zijn met een waarde van 0 tot 15. De kleurcodes hebben de volgende betekenis:

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Als geen kleurcode wordt opgegeven, zal voor <Z> de laatst ingegeven voorgrondkleur worden gebruikt. De "default"-waarde van <Z> (dat wil zeggen de waarde bij het inschakelen van de computer) is 15.

Zie ook PRESET

Voorbeeld

```
10 SCREEN 2:COLOR 15,4,7
20 LINE(40,40)-(215,151),15,B
30 FOR I=0 TO 1000
40 A=INT(RND(1)*173)+41
50 B=INT(RND(1)*109)+41
60 PSET (A,B)
70 FOR K=0 TO 300:NEXT
```

```
80 NEXT I
90 END
RUN
```

114 PUT

Formaat PUT [#]<X>[,<Y>]

Doel Het wegschrijven van een "record" naar een willekeurig toegankelijk bestand op diskette

Categorie Instructie

Opmerkingen <X> is het nummer waaronder het bestand is geopend met de instructie OPEN.

<Y> is het nummer van het "record" (een verzameling bij elkaar behorende gegevens). <Y> moet een integer zijn met een waarde van 0 tot 32767.
Als <Y> niet is opgegeven, wordt de "record" weggeschreven met een nummer dat 1 hoger is dan de laatste ingelezen of weggeschreven record.

In het geheugen moet een buffer worden gereserveerd met de instructie FIELD.

Gebruik de instructie GET om een "record" vanaf diskette in het geheugen van de computer in te lezen.

De instructie PUT kan alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld `zie FIELD`

---

115 PUT SPRITE

Formaat PUT SPRITE <Z>[,[STEP](<X>,<Y>)][,<XX>][,<YY>]

Doel Het op het scherm plaatsen van een gespecificeerde sprite in grafische stand 1 en 2

Categorie Instructie

Opmerkingen <Z> geeft de prioriteit van de sprite aan. <Z> moet een integer zijn met een waarde van 0 tot 31.

<X> is de X-coördinaat van het beeldpunt; <X> moet een integer zijn met een waarde tussen -32 en 255.

<Y> is de Y-coördinaat van het beeldpunt; <Y> moet een integer zijn met een waarde tussen -32 en 191, 208 of 209
Heeft <Y> een waarde van 208, dan zullen alle sprites met een lagere waarde van het scherm verdwijnen.
Heeft <Y> een waarde van 209, dan zal alleen deze sprite van het scherm verdwijnen..

Als het woord STEP wordt gebruikt, gelden de waarden van <X> en <Y> ten opzichte van de cursorpositie. In dit geval mogen <X> en <Y> negatieve integers zijn.

Als <X> en <Y> zijn weggelaten, worden voor X- en Y-coördinaat de dan geldende waarden gebruikt.

<XX> is de kleurcode; <Z> moet een integer zijn met een waarde van 0 tot 15. De kleurcodes hebben de volgende betekenis:

0 = transparant	8 = rood
1 = zwart	9 = lichtrood
2 = groen	10 = donkergeel
3 = lichtgroen	11 = lichtgeel
4 = donkerblauw	12 = donkergroen
5 = lichtblauw	13 = magenta (paars)
6 = donkerrood	14 = grijs
7 = cyaan	15 = wit

Als geen kleurcode wordt opgegeven, zal voor <XX> de laatst ingegeven voorgrondkleur worden gebruikt. De "default"-waarde van <XX> bij het inschakelen van de

computer) is 15.

<YY> is het nummer van de sprite, zoals dat is toegekend met de variabele SPRITE$(<YY>). Is <YY> niet opgegeven, dan is het sprite-nummer gelijk aan het prioriteitsnummer.

De afmetingen van de sprite worden bepaald met de instructie SCREEN

Voorbeeld Zie ON SPRITE GOSUB

---

116 READ

Formaat READ <variabele>[,<variabele>...]

Doel Het lezen van een constante uit een DATA-instructie en het toekennen daarvan aan een variabele

Categorie Instructie

Opmerkingen De instructie READ kan alleen worden gebruikt in combinatie met de instructie DATA. De instructie READ geeft toegang tot de gegevens die zijn vastgelegd met de instructie DATA; dat gebeurt in de volgorde van de programmaregels.

De variabele in een READ-instructie mag zowel een numerieke als een alfanumerieke variabele zijn, op voorwaarde dat het type van de variabele correspondeert met dat van het gelezen gegeven in de desbetreffende DATA-instructie.

Als u probeert met de READ-instructie een groter aantal gegevens in te lezen dan het aantal constanten in de DATA-instructies, zal MSX-BASIC de foutmelding "Out of DATA" op het scherm zetten.

Is het aantal variabelen in de READ-instructie kleiner dan het aantal constanten in de DATA-instructie, dan zal een volgende READ-instructie beginnen met het lezen van de nog niet eerder gelezen constanten in dezelfde DATA-instructie.
Is er geen volgende READ-instructie, dan worden niet-gelezen constanten in de DATA-instructie genegeerd. READ begint met het lezen van de eerste DATA-instructie die in het programma voorkomt, tenzij met de instructie RESTORE een regelnummer wordt opgegeven waar het lezen moet beginnen.

Voorbeeld

```
10 DATA 123,ABC,48.5
20 READ A%,A$,A!
30 PRINT A%,A$,A!
40 RESTORE
50 READ A%:PRINT A%
60 END
RUN
```

```
123          ABC
48.5
123
```

117 REM

Formaat REM <opmerking>

Doel Het opnemen van uitleg in een programma

Categorie Instructie

Opmerkingen REM-instructies worden door MSX-BASIC niet uitgevoerd. Ze worden echter wel op het scherm weergegeven na het commando LIST en afgedrukt na het commando LLIST. REM-instructies dienen uitsluitend om aan het programma een verklaring van de opzet of de werking toe te voegen.

Met de instructies GOTO en GOSUB kan naar een programmaregel met een REM-instructie worden gesprongen. In dat geval gaat het programma verder met het uitvoeren van de eerstvolgende programmaregel die niet met een REM-instructie begint.

MSX-BASIC negeert alle instructies van een programmaregel die na de REM-instructie komen. Wilt u in één regel instructies, die moeten worden uitgevoerd, en REM-instructies opnemen, zet de REM-instructie dan achteraan.

In plaats van het woord REM mag u ook een apostrof (') gebruiken.

Gebruik de instructie REM niet in DATA-instructies.

Voorbeeld 1

```
10 CLS
20 REM Getallen van Fibonacci
30 A=1:B=1
40 PRINT A:PRINT B
50 FOR I=1 TO 18
60 S=A+B:A=B:B=S
70 PRINT S
80 NEXT
```

Voorbeeld 2

```
20 CLS:' Getallen van Fibonacci
30 A=13:B=21
40 PRINT A:PRINT B
50 FOR I=1 TO 18
60 S=A+B:A=B:B=S
70 PRINT S
80 NEXT
```

118 RENUM

Formaat RENUM [[<X>][,<Y>][,<Z>]]

Doel Het hernummeren van programmaregels

Categorie Commando

Opmerkingen <X> is het nummer van de eerste programmaregel volgens de nieuwe nummering. Als <X> niet is opgegeven zal de eerste regel van het hernummerde programma regelnummer 10 hebben.
<Y> is het nummer van de eerste regel waar het hernummeren moet beginnen. Is <Y> niet opgegeven, dan begint het hernummeren bij de eerste programmaregel.
<Z> is de stapgrootte, dat wil zeggen het verschil tussen de nummers van twee opeenvolgende programmaregels. Is <Z> niet opgegeven, dan neemt MSX-BASIC aan dat de stapgrootte 10 moet zijn.
Het commando RENUM hernummert automatisch alle verwijzingen naar regelnummers in een programma die in combinatie worden gebruikt met de instructies GOTO, GOSUB, IF...THEN...ELSE, ON GOTO, ON GOSUB, ON ERROR GOTO en dergelijke. Vindt MSX-BASIC een verwijzing naar een niet bestaand regelnummer, dan verschijnt de foutmelding "Unidentified line <XX> in <YY>" op het scherm, waarin <XX> het niet-bestaande regelnummer is waar naar wordt verwezen en <YY> het nieuwe nummer van de regel waarin die verwijzing voorkomt. In dit geval zal het niet-bestaande regelnummer <XX> in de hernummerde regel <YY> blijven staan.
Het commando RENUM kan niet worden gebruikt om de volgorde van de regelnummers te veranderen.
In voorbeeld 1 zal het hernummerde programma de regelnummers 10, 20, 30 enz. krijgen. In voorbeeld 2 zal het hernummeren beginnen bij de oude regel 900. Deze krijgt nummer 1000. De stapgrootte is 50, dus de volgende regel krijgt nummer 1050. De verwijzingen zullen in alle regels worden aangepast, ook in regels met een nummer lager dan 900.

Voorbeeld 1 `RENUM`

Voorbeeld 2 `RENUM 1000,900,50`

119 RESTORE

Formaat RESTORE [<X>]

Doel Het lezen van constanten uit een DATA-instructie vanaf een gegeven regelnummer door middel van een READ-instructie

Categorie Instructie

Opmerkingen <X> is het regelnummer van de DATA-instructie waar met lezen moet worden begonnen.
Nadat de instructie RESTORE is uitgevoerd, zal de READ-instructie de eerste constante van de opgegeven DATA-regel lezen.

Is <X> niet opgegeven, dan begint het lezen bij de eerste DATA-instructie van het programma.

Voorbeeld

```
10 DATA 123,ABC,48.5
20 READ A%,A$,A!
30 PRINT A%,A$,A!
40 RESTORE
50 READ A%:PRINT A%
60 END
RUN
 123            ABC
 48.5
 123
```

---

120 RIGHT$

Formaat RIGHT$(<X$>,<X>)

Doel Geeft een string, bestaande uit de <X> meest rechtse tekens van <X$>

Categorie Functie

Opmerkingen <X> is een integer tussen 0 en 255.

Als <X> groter is dan de lengte dan <X$>, wordt de gehele inhoud van <X$> gevonden.

Als <X> nul is, is de uitkomst van de functie een lege string.

Voorbeeld

```
10 A$="BASIC"
20 FOR I=1 TO LEN(A$)
30 PRINT RIGHT$(A$,I)
40 NEXT
50 END
RUN
C
IC
SIC
ASIC
BASIC
```

121 RND

Formaat RND(<X>)

Doel Geeft een willekeurig getal met een waarde tussen 0 en 1

Categorie Functie

Opmerkingen Elke keer dat het programma na het commando RUN opnieuw wordt uitgevoerd, wordt dezelfde reeks quasi-willekeurige getallen opgewekt. Om absoluut willekeurige getallen op te wekken, moet de generator elke keer bij het begin van de programma-uitvoering worden "geschud". Dat kan door gebruik te maken van de variabele TIME, zoals in regel 10 van het voorbeeld.

Is <X> groter dan 0, dan geeft de functie RND elke keer het volgende getal uit de reeks.

Is <X> gelijk aan 0, dan wordt het laatste getal uit de reeks telkens herhaald.

Is <X> kleiner dan 0, dan wordt telkens dezelfde reeks getallen opgewekt.

Laat u in het voorbeeld regel 10 weg, dan krijgt u elke keer na RUN dezelfde serie gekleurde ellipsen.

Voorbeeld

```
10 R=RND(-TIME)
20 FOR I=1 TO 10
30 SCREEN 2:COLOR 15,4,7
40 C=INT(RND(1)*15+1)
50 IF C=4 GOTO 40
60 CIRCLE (100,80),50,C
70 PAINT (100,80),C
80 FOR J=0 TO 300:NEXT
90 NEXT I
100 END
RUN
```

122 RSET

Formaat RSET <X$> = <Y$>

Doel De variabele <X$> vullen met de inhoud van de stringvariabele <Y$> of de expressie <Y$>, te beginnen aan de rechterkant van <Y$>

Categorie Instructie

Opmerkingen De instructie LSET of RSET moet worden gebruikt om de gegevens, die naar een willekeurig toegankelijk bestand op diskette moeten worden weggeschreven, in de buffer te plaatsen.

Alvorens numerieke gegevens in de buffer te plaatsen moeten deze in alfanumerieke vorm worden omgezet met een van de functies MKI$, MKS$ of MKD$.

Na het lezen van deze gegevens uit de buffer moeten ze weer van alfanumerieke in numerieke vorm worden terugvertaald met een van de functies CVI, CVS of CVD.

RSET kan alleen worden gebruikt als een of twee diskettestations op de computer zijn aangesloten.

Voorbeeld

```
10 MAXFILES=1
20 OPEN "A:TEST" AS #1
30 FIELD #1,2 AS N1$,4 AS N2$,8 AS
N3$,20 AS N4$
40 INPUT "A%";A%
50 INPUT "B!";B!
60 INPUT "C#";C#
70 INPUT "D$";D$
80 RSET N1$=MKI$(A%):RSET N2$=MKS$
(B!):RSET N3$=MKD$(C#):LSET N4$=D$
90 PUT #1,1
100 A%=0:B!=0:C#=0:D$=""
110 PRINT A%;B!;C#;D$
120 GET #1,1
130 A%=CVI(N1$):B!=CVS(N2$):C#=CVD
(N3$):D$=N4$
140 PRINT A%;B!;C#;D$
150 CLOSE #1
160 END
```

RUN

123 RUN

Formaat RUN [<X>]
RUN ["<randapparaat>:<programmanaam>"]

Doel Het starten van de uitvoering van een programma in het geheugen van de computer of op een cassette of diskette

Categorie Commando

Opmerkingen <randapparaat> kan zijn:

CAS	=	cassette
A	=	diskettestation 1
B	=	diskettestation 2

Het programma moet met het commando SAVE in ASCII-code op cassette of diskette zijn weggeschreven.

<programmanaam> is een alfanumerieke constante die is gespecificeerd toen het programma met SAVE werd weggeschreven op cassette of diskette.

Als "<randapparaat>:<programmanaam>" wordt weggelaten, zal het programma in het geheugen van de computer in uitvoering worden genomen. In dit geval is <X> het regelnummer waar met de uitvoering moet worden begonnen. Is <X> niet opgegeven, dan begint de uitvoering met de eerste regel van het programma.

In voorbeeld 1 begint de uitvoering met de eerste regel van het programma.
In voorbeeld 2 begint de uitvoering op regel 100.
In voorbeeld 3 wordt eerst het programma DEMO van cassette gelezen en onmiddellijk na het inlezen in uitvoering genomen.

RUN CAS kan alleen worden gebruikt als op uw computer een cassetterecorder is aangesloten.

RUN A en RUN B kunnen alleen worden gebruikt als een of twee diskettestations op uw computer zijn aangesloten.

Voorbeeld 1 RUN

Voorbeeld 2 RUN 100

Voorbeeld 3 RUN "CAS:DEMO"

124 SAVE

Formaat SAVE "<randapparaat>:<programmanaam>"

Doel Vanuit het computergeheugen wegschrijven van een programma naar een randapparaat

Categorie Commando

Opmerkingen <randapparaat> kan zijn:

CAS	=	datarecorder
A	=	diskettestation 1
B	=	diskettestation 2

Met het commando SAVE wordt het programma in ASCII-code op de cassette of de diskette weggeschreven.

<programmanaam> is een stringconstante; dezelfde naam moet worden gebruikt als het programma met het commando LOAD of MERGE van cassette of diskette wordt ingelezen.

SAVE CAS kan alleen worden gebruikt als een datarecorder op uw MSX-computer is aangesloten.

SAVE A en SAVE B kunnen alleen worden gebruikt als een of twee diskettestations op uw MSX-computer zijn aangesloten.

Voorbeeld

```
SAVE "CAS:DEMO"
```

---

125 SCREEN

Formaat SCREEN [<X>][,<Y>][,<Z>][,<XX>][,<YY>]

Doel Het kiezen van het schermtype, de grootte van de sprites, het geluid bij het indrukken van een toets, de transmissiesnelheid voor de cassette en het type printer

Categorie Instructie

Opmerkingen <X> is het schermtype en moet een integer zijn met een waarde van 0 tot en met 3. <X> heeft de volgende betekenis;
0 = tekststand 1
1 = tekststand 2
2 = grafische stand 1
3 = grafische stand 2

Is <X> niet opgegeven, dan wordt het laatste schermtype gebruikt. De "default"-waarde van <X> (bij het inschakelen van de computer) is 0.
De grafische instructies PUT SPRITE, CIRCLE, DRAW, LINE, PAINT, PSET, PRESET, ON SPRITE GOSUB, SPRITE ON/OFF/STOP en POINT kunnen alleen in de grafische standen worden gebruikt.

<Y> bepaalt de grootte van de sprites en moet een integer zijn met een waarde van 0 tot en met 3. <Y> heeft de volgende betekenis:
0 = kleine sprites (8 x 8 beeldelementen)
1 = kleine sprites, vergroot tot 16 x 16 beeldelementen
2 = grote sprites (16 x 16 beeldelementen)
3 = grote sprites, vergroot tot 32 x 32 beeldelementen

Is <Y> niet opgegeven, dan wordt de laatst opgegeven sprite-grootte gebruikt.
De "default"-waarde van <Y> (bij het inschakelen van de computer) is 0.

<Z> bepaalt of een klik wordt gehoord bij het indrukken van een toets. <X> moet 0 of 1 zijn en heeft de volgende betekenis:
0 = geen klik
1 = wel een klik
Is <Z> niet opgegeven, dan geldt de laatst opgegeven waarde van <Z>.

De "default"-waarde van <Z> bij het inschakelen van de computer is 1.

<XX> is de "baud rate" voor de datarecorder, dat wil zeggen de snelheid waarmee de gegevens worden ingelezen of weggeschreven. <XX> heeft de waarde 1 of 2, met de volgende betekenis:
1 = 1200 baud (bits per seconde)
2 = 2400 baud
Is <XX> niet opgegeven, dan wordt de laatst gekozen snelheid gebruikt.
De "default"-waarde van de lees- en schrijfsnelheid bij het inschakelen van de computer is 1 (1200 baud).

<YY> biedt de mogelijkheid aan de computer op te geven of al dan niet een speciale MSX-printer wordt gebruikt. <YY> moet 1 of 0 zijn en heeft de volgende betekenis:
0 = MSX-printer
1 = andere printer
Is <YY> niet opgegeven, dan neemt MSX-BASIC aan dat het laatst gekozen printertype wordt gebruikt.
De "default"-waarde van <YY> (bij het inschakelen van de computer) is 0.

Voorbeeld

```
10 FOR I=2 TO 3
20 SCREEN I:COLOR 15,4,7
30 LINE (32,32)-(200,200),6
40 FOR K=0 TO 3000:NEXT
50 NEXT I
60 END
RUN
```

---

126 SGN

Formaat SGN(<X>)

Doel Bepalen of <X> positief, negatief of nul is

Categorie Functie

Opmerkingen De functie SGN geeft als uitkomst 1 als X > 0, 0 als X = 0 en -1 als X < 0.

Voorbeeld

```
10 INPUT "Voer een getal in";N
20 H=SGN(N):H=H+2
30 ON H GOSUB 60,70,80
40 PRINT B$
50 END
60 B$="het getal is negatief.":RETURN
70 B$="het getal is nul":RETURN
80 B$="het getal is positief":RETURN
RUN
```

127 SIN

Formaat SIN(<X>)

Doel Berekenen van de sinus van de hoek <X>, waarbij <X> is uitgedrukt in radialen

Categorie Functie

Opmerkingen Uit onderstaand voorbeeld blijkt dat 355/113 een uitstekende benadering van pi geeft.

Voorbeeld

```
10 PI=3.14
20 PRINT SIN(PI/2)
30 PI=22/7
40 PRINT SIN(PI/2)
50 PI=355/113
60 PRINT SIN(PI/2)
70 END
RUN
 .99999968293185
 .99999980013338
 1
```

---

128 SOUND

Formaat SOUND <X>,<Y>

Doel Het plaatsen van een waarde in een van de registers van de PSG

Categorie Instructie

Opmerkingen <X> is het nummer van het register. <X> moet een integer zijn met een waarde van 0 tot en met 15.

<Y> is de waarde die in register <X> moet worden geplaatst. <Y> moet een integer zijn met een waarde van 0 tot en met 255.

Gebruik deze instructie alleen als u vertrouwd bent met de werking van de programmeerbare geluidsgenerator.

Voorbeeld

```
10 FOR I=0 TO 13
20 SOUND I,0
30 NEXT I
40 SOUND 7,62:SOUND 8,15
50 FOR I=0 TO 255
60 SOUND 0,I
70 NEXT I
80 SOUND 8,0
90 END
RUN
```

129 SPACE$

Formaat SPACE$(<X>)

Doel Het vullen van een string met <X> spaties

Categorie Functie

Opmerkingen <X> moet een integer zijn met een waarde van 0 tot en met 255.

Voorbeeld

```
10 FOR I=0 TO 5
20 A$=SPACE$(I)
30 PRINT A$;I
40 NEXT
50 END
RUN
 0
  1
   2
    3
     4
      5
```

---

130 SPC

Formaat SPC(<X>)

Doel Het afdrukken van <X> spaties op het scherm of op de printer

Categorie Functie

Opmerkingen <X> moet een integer zijn met een waarde van 0 tot en met 255.

Deze functie kan alleen worden gebruikt in combinatie met de instructies PRINT en LPRINT.

Voorbeeld

```
10 PRINT"Hier"SPC(5)"en"SPC(5)"daar"
20 END
RUN
Hier     en     daar
```

131 SPRITE ON/OFF/STOP

Formaat SPRITE ON
SPRITE OFF
SPRITE STOP

Doel Activeren van de controle op het botsen van sprites

Categorie Instructie

Opmerkingen Na de instructie SPRITE ON zal MSX-BASIC bij elke instructie controleren of er een botsing heeft plaatsgevonden tussen twee sprites. Is dat het geval, dan zal MSX-BASIC de subroutine uitvoeren die met de instructie ON SPRITE GOSUB is gegeven.

Na de instructie SPRITE OFF zal MSX-BASIC niet langer controleren of een botsing tussen sprites plaats vindt.

Na de instructie SPRITE STOP zal MSX-BASIC eveneens bij elke instructie controleren of een botsing tussen twee sprites heeft plaatsgevonden. Nu zal echter niet onmiddellijk de subroutine worden uitgevoerd, die met ON SPRITE GOSUB is gegeven.
Daarentegen zal MSX-BASIC onthouden of een botsing heeft plaatsgevonden en de subroutine uitvoeren onmiddellijk nadat de instructie SPRITE ON is gegeven.

Voorbeeld `Zie ON SPRITE GOSUB`

---

132 SPRITE$

Formaat SPRITE$(<X>)

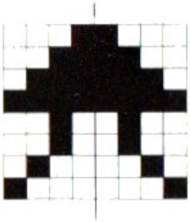
Doel Het definiëren van sprites in de grafische standen 1 en 2

Categorie Systeemvariabele

Opmerkingen <X> is het nummer van de sprite. Als met de instructie SCREEN de grootte van de sprite op 0 of 1 is gezet, moet <X> een integer zijn met een waarde tussen 0 en 255. Is de grootte van de sprites op 2 of 3 gesteld, dan moet <X> een integer zijn met een waarde van 0 tot en met 63.

De inhoud van de variabele SPRITE$ mag een binaire, een hexadecimale of een decimale waarde hebben, zoals in het onderstaande voorbeeld.

In de volgende voorbeelden is de inhoud van de variabele SPRITE$ op drie verschillende manieren (respectievelijk binair, hexadecimaal en decimaal) door middel van DATA-instructies gedefinieerd.



Binary		Hexadecimal		Decimal
&B00011000	=	&H18	=	24
&B00111100	=	&H3C	=	60
&B01111110	=	&H7E	=	126
&B11111111	=	&HFF	=	255
&B00100100	=	&H24	=	36
&B00100100	=	&H24	=	36
&B01000010	=	&H42	=	66
&B10000001	=	&H81	=	129

Voorbeeld 1

```
10 SCREEN 2,0
20 B$=""
30 FOR I=1 TO 8
40 READ A$:B$=B$+CHR$(VAL("&B"+A$))
50 NEXT I
60 SPRITE$(0)=B$
70 DATA 00011000,00111100,01111110,
11111111,00100100,00100100,01000010,
10000001
```

Voorbeeld 2

```
10 SCREEN 2,0
20 B$=""
30 FOR I=1 TO 8
40 READ A$:B$=B$+CHR$(VAL("&H"+A$))
50 NEXT I
60 SPRITE$(0)=B$
70 DATA 18,3C,7E,FF,24,24,42,81
```

Voorbeeld 3

```
10 SCREEN 2,0
20 B$=""
30 FOR I=1 TO 8
40 READ A:B$=B$+CHR$(A)
50 NEXT I
60 SPRITE$(0)=B$
70 DATA 26,60,126,255,36,36,66,129
```

133 SQR

Formaat SQR(<X>)

Doel Het berekenen van de vierkantswortel van <X>

Categorie Functie

Opmerkingen <X> moet gelijk zijn aan of groter zijn dan nul

Voorbeeld

```
10 X=10
20 Y!=SQR(X):PRINT Y!
30 Z=SQR(X):PRINT Z
40 END
RUN
 3.16228
 3.1622776601684
```

134 STICK

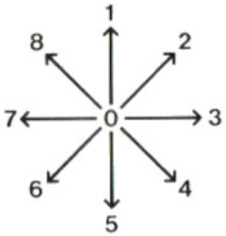
Formaat STICK(<X>)

Doel Het opvragen van de stand van een spelregelaar ("joystick") of de status van de cursor-toetsen

Categorie Functie

Opmerkingen <X> kan een van de volgende betekenissen hebben:
0 = de cursor-besturingstoetsen doen dienst als spelregelaar
1 = heeft betrekking op de spelregelaar, aangesloten op connector 1
2 = heeft betrekking op de spelregelaar, aangesloten op connector 2

Deze functie geeft de volgende waarden:
0 = neutraal
1 = noord
2 = noordoost
3 = oost
4 = zuidoost
5 = zuid
6 = zuidwest
7 = west
8 = noordwest



De even waarden ontstaan als twee naast elkaar gelegen cursortoetsen tegelijk worden ingedrukt.

Voorbeeld

```
10 CLS:PRINT "Druk een cursor-
besturingstoets in"
20 A%=STICK(0):IF A%=0 THEN 20 ELSE
PRINT A%" ";
30 FOR I=0 TO 300:NEXT
```

135 STOP

Formaat STOP

Doel Het onderbreken van de uitvoering van een programma

Categorie Instructie

Opmerkingen De instructie STOP kan op elke plaats in een programma worden gebruikt.

Als MSX-BASIC een STOP-instructie tegenkomt, zal de uitvoering van het programma worden onderbroken en verschijnt de melding "Break in xxxxx" op het scherm, waarin "xxxxx" het regelnummer is waarin het programma werd onderbroken.

De uitvoering kan worden hervat met het commando CONT.

Voorbeeld

```
10 PRINT "TEST"
20 STOP
30 PRINT "Verder na CONT"
40 END
RUN
TEST
Break in 20
Ok
CONT (zelf intoetsen in de directe
stand)
Verder na CONT
```

---

136 STOP ON/OFF/STOP

Formaat
STOP ON
STOP OFF
STOP STOP

Doel
Activeren van de controle op het gelijktijdig indrukken van de toetsen CTRL en STOP

Categorie
Instructie

Opmerkingen
<X> is het nummer van de functietoets (1...10).

Na de instructie STOP ON zal MSX-BASIC bij elke instructie controleren of de toetsen CTRL en STOP tegelijk zijn ingedrukt. Is dat het geval, dan zal MSX-BASIC naar de subroutine gaan die met de instructie ON STOP GOSUB is gegeven.

Na de instructie STOP OFF zal MSX-BASIC niet langer controleren of de beide toetsen zijn ingedrukt.

Na de instructie STOP STOP zal MSX-BASIC eveneens bij elke instructie controleren of de beide toetsen zijn ingedrukt. Nu zal echter niet onmiddellijk de subroutine worden uitgevoerd, die met ON STOP GOSUB is gegeven.
Daarentegen zal MSX-BASIC onthouden of de toetsen CTRL en STOP tegelijk ingedrukt zijn geweest en de subroutine uitvoeren onmiddellijk nadat de instructie STOP ON is gegeven.

Voorbeeld

```
10 ON STOP GOSUB 50
20 STOP ON
30 INPUT A$
40 IF A$="Einde" THEN STOP OFF:END ELSE
GOTO 30
50 PRINT "Tik 'Einde' in; druk daarna
op RETURN":RETURN
RUN
```

137 STRIG

Formaat STRIG(<X>)

Doel Het opvragen van de status van de spatiebalk of de actietoetsen van de spelregelaars

Categorie Functie

Opmerkingen <X> is een integer met een waarde van 0 tot en met 4.
<X> heeft de volgende betekenis:
0 = spatiebalk
1 = actietoets van de spelregelaar, verbonden met aansluiting 1
2 = actietoets van de spelregelaar, verbonden met aansluiting 2
3 = actietoets van de spelregelaar, verbonden met aansluiting 1
4 = actietoets van de spelregelaar, verbonden met aansluiting 2

De functie levert de waarde -1 op als de spatiebalk of een van de actietoetsen van de spelregelaars wordt ingedrukt. Anders is de waarde 0.

Voorbeeld

```
10 PRINT "Druk de spatiebalk in"
20 P=STRIG(0)
30 IF P=-1 THEN BEEP
40 GOTO 20
RUN
```

138 STRIG(X) ON/OFF/STOP

Formaat STRIG(<X>) ON
STRIG(<X>) OFF
STRIG(<X>) STOP

Doel Activeren van de controle op het indrukken van een actietoets of de spatiebalk

Categorie Instructie

Opmerkingen <X> is een integer met een waarde van 0 tot en met 4.
<X> heeft de volgende betekenis:
0 = spatiebalk
1 = actietoets van de spelregelaar, verbonden met aansluiting 1
2 = actietoets van de spelregelaar, verbonden met aansluiting 2
3 = actietoets van de spelregelaar, verbonden met aansluiting 1
4 = actietoets van de spelregelaar, verbonden met aansluiting 2

Na de instructie STRIG(X) ON zal MSX-BASIC bij elke instructie controleren of een actietoets of de spatiebalk is ingedrukt. Is dat het geval, dan zal MSX-BASIC naar de subroutine gaan die met de instructie ON STRIG GOSUB is gegeven.

Na de instructie STRIG(X) OFF zal MSX-BASIC niet langer controleren of de spatiebalk of een actietoets is ingedrukt.

Na de instructie STRIG(X) STOP zal MSX-BASIC eveneens bij elke instructie controleren of de spatiebalk of een actietoets is ingedrukt. Nu zal echter niet onmiddellijk de subroutine worden uitgevoerd, die met ON STRIG GOSUB is gegeven.
Daarentegen zal MSX-BASIC onthouden of een actietoets of spatiebalk ingedrukt is geweest en de subroutine uitvoeren onmiddellijk nadat de instructie STRIG(X) ON is gegeven.

Voorbeeld Zie ON STRIG GOSUB

139 STR$

Formaat STR$(<X>)

Doel Het geven van een alfanumerieke representatie van de numerieke expressie <X>

Categorie Functie

Opmerkingen Gebruik de functie VAL om een alfanumerieke waarde om te zetten in een numerieke representatie.

Voorbeeld

```
10 HOEVEELHEID=12.34216:PRINT HOEVEEL
HEID
20 A$=STR$(HOEVEELHEID)
30 H%=INSTR(A$,".")
40 IF H%=<1 GOTO 80
50 MID$(A$,H%,1)=","
60 H$=LEFT$(A$,H%+2)
70 PRINT H$
80 END
RUN
 12.34216
 12,34
```

---

140 STRING$

Formaat

STRING$(<X>,<Y>)
STRING$(<X>,<Y$>)

Doel

Het produceren van een string met lengte <X>, geheel bestaande uit tekens met tekencode <Y> of het eerste teken van de string <Y$>

Categorie

Functie

Opmerkingen

<X> is het aantal tekens van de string. <X> moet een integer zijn met een waarde van 0 tot en met 255.

<Y> is de code van het teken dat moet worden afgedrukt. <Y> moet een integer zijn met een waarde van 32 tot en met 255.

<Y$> is een alfanumerieke variabele. Wordt voor <Y$> een alfanumerieke constante gebruikt, dat wil zeggen wordt voor <Y$> in de instructie STRING$ een alfanumeriek teken ingevuld, dan moet <Y$> tussen aanhalingstekens staan.

Voorbeeld 1

```
10 SCREEN1:COLOR 15,4,7
20 A$=CHR$(42)
30 FOR K=1 TO 20
40 C=INT(RND(1)*20)
50 LOCATE 3,K:PRINT USING"##";C
60 LOCATE 6,K:PRINT STRING$(C,A$)
70 NEXT
80 END
RUN
```

Voorbeeld 2

```
10 SCREEN1:COLOR 15,4,7
20 A$="*"
30 FOR K=1 TO 20
40 C=INT(RND(1)*20)
50 LOCATE 3,K:PRINT USING"##";C
60 LOCATE 6,K:PRINT STRING$(C,A$)
70 NEXT
80 END
RUN
```

Voorbeeld 3

```
10 SCREEN1:COLOR 15,4,7
30 FOR K=1 TO 20
40 C=INT(RND(1)*20)
50 LOCATE 3,K:PRINT USING"##";C
60 LOCATE 6,K:PRINT STRING$(C,42)
70 NEXT
80 END
RUN
```

Voorbeeld 4

```
10 SCREEN1:COLOR 15,4,7
30 FOR K=1 TO 20
40 C=INT(RND(1)*20)
50 LOCATE 3,K:PRINT USING"##";C
60 LOCATE 6,K:PRINT STRING$(C,"*")
70 NEXT
80 END
RUN
```

141 SWAP

Formaat SWAP <variabele>,<variabele>

Doel Het verwisselen van de inhoud van twee variabelen

Categorie Instructie

Opmerkingen De instructie SWAP kan worden gebruikt voor het verwisselen van de inhoud van twee variabelen. Dit mogen variabelen van elk type zijn (integer-, enkele-precisie-, dubbele-precisie- en stringvariabelen), op voorwaarde dat de twee van hetzelfde type zijn.

Voorbeeld 1

```
10 X=3:Y=7
20 GOSUB 60
30 SWAP X,Y
40 GOSUB 60
50 END
60 PRINT "X="X"Y="Y
70 RETURN
RUN
X= 3 Y= 7
X= 7 Y= 3
```

Voorbeeld 2

```
10 X$="een":Y$="allen"
20 GOSUB 60
30 SWAP X$,Y$
40 GOSUB 60
50 END
60 PRINT X$" voor "Y$
70 RETURN
RUN
een voor allen
allen voor een
```

142 TAB

Formaat TAB(<X>)

Doel De cursor naar een gegeven positie <X> op dezelfde regel verplaatsen

Categorie Functie

Opmerkingen De functie TAB mag alleen worden gebruikt in combinatie met de instructie PRINT of LPRINT.

<X> moet een integer zijn met een waarde van 0 tot en met 255. Is <X> groter dan het aantal posities van een schermregel of een printerregel, dan gaat de cursor automatisch verder op de volgende regel.

Is de cursor al voorbij positie <X>, dan wordt de functie TAB genegeerd.

Voorbeeld

```
10 SCREEN 0:WIDTH 40:COLOR 15,1:CLS
20 FOR I=0 TO 19 STEP 4
30 PRINT TAB(I)"MSX"
40 NEXT
50 END
RUN
MSX
    MSX
        MSX
            MSX
                MSX
```

---

143 TAN

Formaat TAN(<X>)

Doel Berekenen van de tangens van de hoek <X>, waarbij <X> is uitgedrukt in radialen

Categorie Functie

Opmerkingen Uit onderstaand voorbeeld blijkt dat 355/113 een uitstekende benadering van pi geeft.

Voorbeeld

```
10 PI=3.14
20 PRINT USING"#.#######";TAN(PI*2)
30 PI=22/7
40 PRINT USING"#.#######";TAN(PI*2)
50 PI=355/113
60 PRINT USING"#.#######";TAN(PI*2)
70 END
RUN
-.0031853
0.0025290
0.0000005
```

144 TIME

Formaat TIME

Doel Uitlezen van de ingebouwde klok

Categorie Systeemvariabele

Opmerkingen Elke keer dat de VDP een interruptie genereert (50 keer per seconde) wordt de variabele TIME met 1 verhoogd.

Als geen interrupties worden gegenereerd (bij voorbeeld als een programma wordt weggeschreven naar of wordt ingelezen van cassette) blijft de variabele TIME onveranderd.

Voorbeeld

```
10 SCREEN 0:WIDTH 40:CLS
20 LOCATE 26,1
30 PRINT"uur:min:sec"
40 TIME=0:T=0:W=0:WV=0
50 W=TIME/50
60 IF W<WV then T=T+1
70 WV=W:W=W+T*1311
80 H=INT(W/3600) MOD 24:IF INT(W/
3600)>24 THEN 40
90 M=INT(W/60) MOD 60
100 S=W-60*INT(W/60)
110 LOCATE 26,2
120 PRINT USING"###:###:###";H;M;S
130 GOTO 50
```

145 TROFF

Formaat TROFF

Doel Het beëindigen van de foutopsporingsmethode

Categorie Commando

Opmerkingen De "trace"-functie, in werking gesteld met het commando TRON, wordt beëindigd met het commando TROFF.

De "trace"-functie wordt ook beëindigd met het commando NEW, maar daarmee wordt tevens het programma in het geheugen gewist.

Voorbeeld `Zie TRON`

---

146 TRON

Formaat TRON

Doel Het activeren van de "trace"-functie

Categorie Commando

Opmerkingen Als het commando TRON is ingevoerd, drukt MSX-BASIC tussen rechte haken de nummers van de regels af die hij uitvoert. Deze methode is een handig hulpmiddel om fouten in een programma op te sporen (Engels: to trace).

Voorbeeld

```
10 FOR I01 TO 3
20 PRINT I
30 NEXT
40 END
TRON
Ok
RUN
[10][20] 1
[30][20] 2
[30][20] 3
[30][40]
Ok
TROFF
Ok
```

---

147 USR

Formaat USR[<X>](<Y>)

Doel Het aanroepen van een machinetaalroutine

Categorie Functie

Opmerkingen <X> is een integer met een waarde van 0 tot en met 9. <X> is het nummer van de machinetaalroutine die wordt aangeroepen. Als <X> niet is opgegeven, neemt MSX-BASIC aan dat USR0 (nul) is bedoeld.

Het adres waar de uitvoering van de machinetaalroutine moet beginnen dient te worden gedefinieerd met de instructie DEFUSR voordat de routine kan worden aangeroepen; anders volgt de foutmelding "Illegal function call".

<Y> is de waarde die aan de machinetaalroutine moet worden meegegeven. Dit kan, afhankelijk van wat de machinetaalroutine verwacht, een numerieke of een alfanumerieke waarde zijn. De volgende mogelijkheden doen zich voor:

1 Alfanumerieke waarden
Geheugenadres &HF663 bevat de waarde 3. Geheugenadressen &HF7F8 en &HF7F9 bevatten het adres van de zogenaamde stringdescriptor. Deze stringdescriptor heeft een lengte van 3 bytes en beschrijft de eigenlijke string. Het eerste byte geeft de lengte van de string aan; het tweede en derde byte geven het geheugenadres aan waar de string zelf is opgeborgen.

2 Integers
Geheugenadres &HF663 bevat de waarde 2. Geheugenadressen &HF7F8 en &HF7F9 bevatten de inhoud van de integer.

3 Enkele-precisiewaarden
Geheugenadres &HF663 bevat de waarde 4. Geheugenadressen &HF7F6 tot en met &HF7F9 bevatten de inhoud van het enkele-precisiegetal.

4 Dubbele-precisiewaarden
Geheugenadres &HF663 bevat de waarde 8. Geheuge-

nadressen &HF7F6 tot en met &HF7FD bevatten de inhoud van het dubbele-precisiegetal.

De waarden die de machinetaalroutine teruggeeft aan MSX-BASIC moeten door die routine op dezelfde manier in het geheugen worden geplaatst als hierboven is beschreven.

Gebruik de instructie CLEAR om geheugenruimte te reserveren voor uw eigen machinetaalroutines.

Voorbeeld

```
10 CLEAR 200,&HEFFF
20 AB=&HF000
30 FOR I=AB TO AB+9
40 READ A$:A=VAL("&H"+A$)
50 POKE I,A
60 NEXT
70 DEFUSR=&HF000
80 INPUT"Voer een geheel getal in";A%
90 PRINT"Het getal is"A%
100 R=USR(A%)
110 PRINT"De uitkomst is geheel getal
plus 1";R
120 END
130 DATA 23,23,4E,23,46,03,70,2B,71,C9
```

148 VAL

Formaat VAL(<X$>)

Doel Berekenen van de numerieke waarde van de alfanumerieke expressie <X$>

Categorie Functie

Opmerkingen De functie negeert spaties en besturingstekens bij het berekenen van de waarde van een string.

De functie wordt voornamelijk gebruikt om binaire, octale en hexadecimale waarden (die als string worden beschouwd) om te zetten in een numerieke waarde.

Gebruik de functie STR$ om een numerieke waarde om te zetten in een string.

Voorbeeld

```
10 HOEVEELHEID=12.34216:PRINT HOEVEEL
HEID
20 A$=STR$(HOEVEELHEID)
30 H%=INSTR(A$,".")
40 IF H%<1 GOTO 80
50 H%=H%+2
60 B$=LEFT$(A$,H%)
70 HOEVEELHEID=VAL(B$):PRINT HOEVEEL
HEID
80 END
RUN
 12.34216
 12.34
```

149 VARPTR (1)

Formaat VARPTR(<variabele>)

Doel Opvragen van het eerste geheugenadres van een variabele

Categorie Functie

Opmerkingen De functie VARPTR geeft het eerste geheugenadres waar een variabele is opgeslagen. Aan deze variabele moet eerst een waarde zijn toegekend, anders geeft MSX-BASIC de foutmelding "Illegal function call".

Met de functie VARPTR kunnen alle typen variabelen worden gebruikt: integers, enkele- en dubbele-precisievariabelen en strings, ook als deze met de instructie DIM zijn gedimensioneerd.

Aan alle niet gedimensioneerde variabelen moet een waarde zijn toegekend voordat met de functie VARPTR het beginadres van een gedimensioneerde variabele wordt opgevraagd. De reden hiervoor is dat de adressen van de gedimensioneerde variabelen veranderen zodra aan een niet gedimensioneerde variabele een waarde wordt toegekend.

De functie VARPTR wordt meestal gebruikt om het geheugenadres op te vragen van een variabele die moet worden overgebracht naar een machinetaalroutine.

De uitkomst van de functie VARPTR is een getal met een waarde van -32768 tot +32767. Als een negatief getal wordt gevonden, moet hier 65536 bij worden opgeteld om het juiste adres te vinden.

Voorbeeld

```
10 A=10
20 B=VARPTR(A)
30 IF B<0 THEN B=B+65536
40 C$="0000"+HEX$(B)
50 PRINT RIGHT$(C$,4)
60 END
RUN
```

---

150 VARPTR (2)

Formaat VARPTR(# <X>)

Doel Opvragen van het eerste byte van een "file control block" van een bestand

Categorie Functie

Opmerkingen De functie VARPTR geeft het eerste geheugenadres van het "file control block" van een bestand. Het "file control block" is een aantal geheugenplaatsen waarin gegevens worden opgeslagen die nodig zijn voor het beheer van het bestand.

<X> is het nummer waaronder het bestand met de instructie OPEN is geopend.

De uitkomst van de functie VARPTR is een getal met een waarde van -32768 tot +32767. Als een negatief getal wordt gevonden, moet hier 65536 bij worden opgeteld om het juiste adres te vinden.

Voorbeeld

```
10 MAXFILES=2
20 OPEN "CRT:" FOR OUTPUT AS #1
30 OPEN "CAS:DEMO" FOR OUTPUT AS #2
40 A=VARPTR(#1)
50 B=VARPTR(#2)
60 IF A<0 THEN A=A+65536
70 IF B<0 THEN B=B+65536
80 C$="0000"+HEX$(A)
90 PRINT RIGHT$(C$,4)
100 C$="0000"+HEX$(B)
110 PRINT RIGHT$(C$,4)
120 CLOSE
130 END
RUN
```

151 VDP

Formaat VDP(<X>)

Doel Opvragen van de inhoud van de VDP-registers

Categorie Systeemvariabele

Opmerkingen <X> is het nummer van het register. <X> moet een integer zijn met een waarde van 0 tot en met 8.

De inhoud van register 8 mag niet worden veranderd, maar wel worden opgevraagd.

Gebruik de systeemvariabele VDP alleen als u goed op de hoogte bent met de werking van de Video Display Processor (VDP). U kunt dan de inhoud van de registers veranderen met VDP(<X>)=&Hnn of met VDP(<X>)=&Bnnnnnnnn.

Voorbeeld

```
10 FOR I=0 TO 8
20 A=VDP(I)
30 B$="00000000"+BIN$(A)
40 C$="0000"+HEX$(A)
50 PRINT I"  "RIGHT$(B$,8)"  "RIGHT
$(C$,4)
60 NEXT
70 END
```

152 VPEEK

Formaat VPEEK(<X>)

Doel Het opvragen van een byte uit het videogeheugen

Categorie Functie

Opmerkingen Deze functie geeft de decimale waarde van de inhoud van een positie in het videogeheugen. Deze inhoud heeft een waarde van 0 tot en met 255.

<X> is het adres van het videogeheugen waarvan u de waarde wilt opvragen. <X> moet liggen tussen 0 en 16383.

De instructie VPOKE wordt gebruikt om de inhoud van een adres van het videogeheugen te vullen met een bepaalde waarde.

Gebruik deze functie alleen als u goed op de hoogte bent met de werking van de Video Display Processor (VDP).

Voorbeeld

```
10 SCREEN 0:WIDTH 40:CLS
20 FOR I=0 TO 199
30 VPOKE I,32+I
40 NEXT
50 LOCATE 0,6
60 FOR I=0 TO 199
70 A=VPEEK(I)
80 A$=HEX$(A)
90 PRINT A$" ";
100 NEXT
110 END
```

153 VPOKE

Formaat VPOKE <X>,<Y>

Doel Het vullen van een positie van het videogeheugen met een gegeven waarde

Categorie Instructie

Opmerkingen <X> is het adres van het videogeheugen waarvan u de waarde wilt veranderen. <X> moet liggen tussen 0 en 16383.

<Y> is de waarde die op het adres <X> moet worden geplaatst. <Y> moet een waarde hebben van 0 tot en met 255.

De functie VPEEK wordt gebruikt om de inhoud van een adres van het videogeheugen op te vragen.

Gebruik deze instructie alleen als u goed op de hoogte bent met de werking van de Video Display Processor (VDP).

Voorbeeld

```
10 SCREEN 0:WIDTH 40:CLS
20 FOR I=0 TO 199
30 VPOKE I,32+I
40 NEXT
50 LOCATE 0,6
60 FOR I=0 TO 199
70 A=VPEEK(I)
80 A$=HEX$(A)
90 PRINT A$" ";
100 NEXT
110 END
```

154 WAIT

Formaat WAIT <X>,<Y>[,<Z>]

Doel Het opvragen en vergelijken van de status van machine-ingangspoort <X>

Categorie Instructie

Opmerkingen Tijdens de uitvoering van deze instructie wordt de status van ingangspoort <X> vergeleken met de integer expressie <Y>, op basis van XOR, en met de integer expressie <Z>, op basis van AND. Is het resultaat van deze logische vergelijking nul, dan zal MSX-BASIC de status van ingangspoort <X> opnieuw vergelijken. In alle andere gevallen zal het programma worden voortgezet.

Als <Z> niet is opgegeven, wordt automatisch aangenomen dat <Z> gelijk is aan nul.

Voorbeeld `Zie INP en OUT`

---

155 WIDTH

Formaat WIDTH <X>

Doel Instellen van het aantal posities per regel op het scherm

Categorie Instructie

Opmerkingen <X> is het aantal posities per regel. <X> moet een integer zijn met een waarde van 1 tot en met 40 in tekststand 1 en van 1 tot en met 32 in tekststand 2.

Bij het inschakelen van de computer zal MSX-BASIC het aantal posities in tekststand 1 instellen alsof de instructie WIDTH 37 is gegeven. In tekststand 2 is dat WIDTH 29.

Voorbeeld

```
10 SCREEN 0
20 FOR I=37 TO 10 STEP -1
30 WIDTH I
40 PRINT"Dit is een tekstregel op het
scherm"
50 FOR J=1 TO 800:NEXT
60 NEXT
70 END
RUN
```

PHILIPS